NASA TECHNICAL NOTE

NASA TN D-5642

NASA TN D-5642

AN ADAPTIVE RANDOM-SEARCH ALGORITHM FOR IMPLEMENTATION OF THE MAXIMUM PRINCIPLE

by Elwood C. Stewart, William P. Kavanaugh, and David H. Brocker

Ames Research Center
Moffett Field, Calif.

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • FEBRUARY 1970

1. Report No. NASA TN D-5642	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle AN ADAPTIVE RANDOM-SEARCH ALGORITHM FOR IMPLEMENTATION OF THE MAXIMUM PRINCIPLE		5. Report Date February 1970 6. Performing Organization Code
7. Author(s) Elwood C. Stewart, William P. Kavanaugh, and David H. Brocker		8. Performing Organization Report No. A-3134
9. Performing Organization Name and Address NASA Ames Research Center Moffett Field, Calif. 94035		10. Work Unit No. 125-19-04-06-00-21 11. Contract or Grant No. 13. Type of Report and Period Covered TECHNICAL NOTE
12. Sponsoring Agency Name and Address NATIONAL AERONAUTICS AND SPACE ADMINISTRATION Washington, D. C., 20546		14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

In this paper a feasible method of implementing the Maximum Principle is given which can be used to generate explicit optimal solutions to high-order nonlinear control problems, either variable or fixed time. The method is based on an adaptive random-search algorithm, and has a number of advantages over other approaches. Example applications are given for which optimum solutions have not been previously obtained; it is demonstrated that striking improvement in performance can be obtained by the use of optimal nonlinear control.

17. Key Words Suggested by Authors	18. Distribution Statement
Optimal control Maximum Principle Pontryagin Maximum Principle Random search Hybrid computer Satellite attitude control Orbit transfer Search algorithm	Unclassified - Unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	69	$3.00

*For sale by the Clearinghouse for Federal Scientific and Technical Information
Springfield, Virginia 22151

TABLE OF CONTENTS

AN ADAPTIVE RANDOM-SEARCH ALGORITHM FOR IMPLEMENTATION OF THE MAXIMUM PRINCIPLE

By Elwood C. Stewart, William P. Kavanaugh, and David H. Brocker

Ames Research Center

SUMMARY

In this paper a feasible method of implementing the Maximum Principle is given which can be used to generate explicit optimal solutions to complex nonlinear control problems. The method is based on an adaptive random-search algorithm that utilizes direct measurements of hypersurfaces, the analytic forms of which are unknown, to solve a two-point boundary-value problem. The purpose of the search is to locate a set of parameters that correspond to the minimum of the hypersurfaces. The method has a much wider scope of applicability than to the two-point boundary-value problem to which this paper is addressed. No restrictions are placed on the continuity of the surfaces or on the number of maxima or minima. The adaptation enhances convergence by varying the mean and variance of a probability distribution as a function of past performance. The algorithm has local and global search properties so that "hanging up" on local minima or maxima is avoided.

A hybrid computer implementation of the algorithm is discussed, which treats both fixed- and free-time problems. The usefulness of the algorithm is investigated experimentally for a fixed-time fifth-order nonlinear minimum-fuel orbit-transfer problem and for a free-time, sixth-order, nonlinear minimum-fuel satellite acquisition problem. Solutions are obtained for these problems in a wide variety of situations, and system performance in some of these situations improved remarkably. This is directly attributable to the computational method of this paper, which permits one to investigate nonlinear problems in a manner heretofore unattainable. Convergence is generally obtained within several thousand iterations (one to two minutes). Details are given on the manner in which the system iterates, on typical solutions that were obtained for a wide range of situations, and the convergence properties of several variations of the basic algorithm. Cross sections through the boundary hypersurfaces reveal the striking irregularities that high-order systems can have; it has been demonstrated that the adaptive random-search approach is effective in coping with these irregularities.

INTRODUCTION

The Pontryagin Maximum Principle is an exceedingly elegant and powerful theory for determining the optimal control of dynamic systems describable by nonlinear differential equations with bounded control. Although there has been a flurry of activity for several years in its application to low-order

systems, there have been few applications to high-order systems. For high-order systems the Maximum Principle yields much information about the nature of the solution, but the actual solution is generally difficult to obtain. The reason for this is that a difficult, mixed boundary-value problem is invariably encountered, one in which the known boundary conditions are divided between initial and terminal values. Furthermore, the mapping from the initial to terminal boundary values is not generally explicitly known. An equivalent formulation may be employed to transform the mixed boundary-value problem to an initial-value problem by constructing this mapping with the aid of computers. Then the problem can be interpreted as one of determining the set of initial conditions that correspond to a minimum of a hypersurface, this minimum value being known a priori.

One possible approach to the above problem is based on some form of the gradient method. However, there are several disadvantages in this approach: (1) only local properties are utilized, (2) only one local minimum is implicitly assumed, and (3) the gradient is not usually analytically available. These difficulties lead to convergence problems as shown in reference 1. Thus, the approach is of limited value when the surface being studied is multipeaked, discontinuous, or very flat in certain regions. Random-search methods seem promising in overcoming some of the above deficiencies. References 2-4 give good general discussions of the merits of random techniques. Such methods have been used mostly in connection with direct-search problems (refs. 5 and 6). However, in a recent study (ref. 7), a fixed-step, random-sign-type search was used to implement the Maximum Principle, and application was made to linear second- and third-order systems.

A more general approach will be discussed in this paper based on an adaptive random-search method for solving the two-point boundary-value problem involved in the Maximum Principle. The random search has both local and global properties and the method does not depend on the continuity of the surface being searched or the number of minima, and does not require that an analytical expression for the surface be known. The method, which is based on two previous studies (refs. 8 and 9), permits the much larger class of free-time problems to be treated.

It is worth noting that the adaptive random-search method has wider applicability than merely solving problems that result from application of the Maximum Principle and its associated boundary-value problem. That is, the same method could be useful in the more general problem of determining the minimum of an arbitrary hypersurface; much literature is concerned with this more general problem.

Although the approach described in this paper is closely related to such studies, there are significant differences. First, a fairly general set of heuristics is implemented in the present study to enable an accurate but complex model of the problem to be solved. This contrasts with other approaches in which heuristics are used to reduce the problem to a size more suitable to more rigorous mathematical analysis; the latter approach tends to oversimplify the true nature of the surface. A second difference is that general global minimization problems usually require complex and somewhat arbitrary

stopping rules. In the problem presented in this paper, the stopping rule is simple and precise because the minimum value of the surface is known by construction. Finally, the technique of implementation assumes a more prominent position in this study than in the more general studies because the method is dependent on new advances in computer capabilities.

In the sections to follow, the adaptive random-search algorithm for both the fixed-time and free-time situations are described, the respective hybrid computer realizations are discussed, and applications of the method to a fixed-time, fifth-order, nonlinear orbit-transfer problem and to a free-time, sixth-order, nonlinear satellite attitude acquisition problem are presented.

PROBLEM FORMULATION

Although the adaptive random-search method to be discussed in this paper is applicable to a wide range of engineering problems, attention will center on those problems to which the Maximum Principle is applicable. Familiarity with the Maximum Principle is assumed (refs. 5, 6, 7). Thus, we will not review the derivation, the theorems, the assumptions involved, or the boundary-value theory. For purposes of notation, a few remarks are appropriate, however.

The system to be controlled is defined by the vector differential equation

$$\dot{x} = f(x,u,t) \tag{1}$$

where the state vector is $x = (x_1,x_2, \ldots,x_n)$, the control vector is $u = (u_1,u_2, \ldots,u_m)$, f is a vector function, $f = (f_1,f_2, \ldots,f_n)$, and $u \varepsilon U$, the allowable control region. Discussion will center on fixed-time problems because of the convenience in presentation; however, free-time problems will be included, and example problems given for each situation. It is desired to take the system from a given state $x(0)$ to a final target set S so as to minimize the generalized cost function

$$C = \sum_{i=o}^{n} \alpha_i x_i(t) \tag{2}$$

where $x_o(t)$ is the auxiliary state associated with the quantity to be minimized (ref. 7). Some of the possible target sets are: (a) $S \varepsilon R_n$, (b) $S \subset R_n$, and (c) $S = x_f \varepsilon R_n$, corresponding to a free point, a subset of the whole space R_n, and a fixed point.

The application of the Maximum Principle to the stated problem invariably requires the solution to the set of equations:

$$\left.\begin{aligned} \dot{x} &= f(x,u,t) \\ \dot{p} &= g(p,x,u,t) \\ u &= u(x,p) \end{aligned}\right\} \tag{3}$$

subject to certain boundary conditions, and, where p is the adjoint vector, $p = (p_0, p_1, \ldots, p_n)$.

Although the above set of equations obtained from the Maximum Principle contain a good deal of information about the nature of the optimum control, it is generally difficult to obtain an explicit solution that satisfies the desired boundary conditions; that is, we know the control function $u(x,p)$ and the equations for x and p, but at no time do we know the specific values of both x and p. For example, at the initial time, $x(0)$ is generally known from the problem specifications, whereas $p(0)$ is not known. The boundary conditions remaining at the end time will be determined in part by the specified target set and in part by the transversality condition. Thus, there is difficulty in solving these equations even numerically, because the known boundary conditions are split between initial and final times.

THE ADAPTIVE RANDOM-SEARCH ALGORITHM

The general approach used here is based on an adaptive random-search procedure and it obtains explicit solutions for problems to which the Maximum Principle has been applied. To describe the approach it is expedient to discuss (1) the basic search algorithm for fixed-time problems; (2) the changes required for free-time problems; and (3) the adaptive characteristics of the algorithm.

Nonadaptive Random-Search Algorithm for Fixed Time

In this section fixed-time problems in which the system is either autonomous or nonautonomous will be treated. A hybrid computer diagram of the algorithm is illustrated in figure 1. The left half of the figure, indicated by analog computation, is an implementation of the set of equations (3). The right half, indicated by digital computation, is an implementation of those operations that are best done digitally in order to satisfy the boundary conditions involved in the Maximum Principle.

Let us discuss the algorithm in detail. The basic notion in the random-search algorithm is to select the initial condition vector for the adjoint equations from a noise source, which has in this case a gaussian distribution with (for purposes of the immediate discussion) fixed mean m and fixed standard deviation σ. Conceptually, as indicated in the figure, the mean and standard deviations can be separated by adding the m to the output of a gaussian noise source with standard deviation σ and zero mean. Thus on any kth iteration, the vector p^k is

$$p^k = m^k + \xi^k \tag{4}$$

The gaussian noise source generates a purely random, n-dimensional vector sequence $\{\xi^k\}$ with zero mean and independent components:

$$m \equiv E(\xi^k) = 0 \tag{5}$$

$$\text{cov}(\xi^j, \xi^k) = Q = \sigma^2 \delta_{jk} \tag{6}$$

where $\xi = (\xi_1, \xi_2, \ldots, \xi_n)$, and δ_{jk} is the Kronecker delta.

Continuing around the loop in figure 1, the value of p^k as determined by equation (4) becomes $p(0)$, the initial condition for the adjoint equation. Since the value of $p(0)$ together with the given $x(0)$ is sufficient to define a solution of the set (3), these equations represented by the left half of figure 1 can be integrated to the terminal time T. The final state $x^k(T)$ actually achieved generally fails to satisfy the desired terminal state as defined by the target set. Similarly, the final values of the adjoint variables $p^k(T)$ fail to satisfy the boundary conditions required by transversality. For this reason a function to measure the difference between the actual boundary values and the desired boundary values is introduced herein. The deficiencies pointed out in reference 13 of the scalar-valued metric will be avoided. Instead, systems will be partially ordered by a vector-valued metric. Generally, three distinct types of conditions will be satisfied at the terminal time, that due to the displacement components of $x(T)$, that due to the velocity components of $x(T)$, and that due to the transversality relation. Hence, the vector metric will be taken to be of the form

$$J = (J_D, J_V, J_P) \tag{7}$$

where the components refer to displacement, velocity, and transversality (adjoint variables p) errors, respectively. For two systems Z and Z' we will say $Z > Z'$ if and only if $J > J'$; that is, $J_D > J_D'$, $J_V > J_V'$, and $J_P > J_P'$. With this concept one is concerned with choosing from a certain set of systems a noninferior system rather than an optimum system. This concept corresponds more closely with a realistic objective as pointed out in reference 13.

Assume now that the search is purely random with no adaptive characteristics. In this case the dashed lines in figure 1 would be absent. Then as a result of the random vector sequence $\{\xi^k\}$, the vector sequence $\{J^k\}$ is generated. Since the search is purely random, the algorithm logic decides at each iteration whether the value of J has been reduced to zero.

Satisfying the boundary conditions is slightly more involved than the above because in an experimental study it is not likely that the boundary conditions for x and p will ever be precisely achieved; that is, J will never be exactly zero. For this reason we will enlarge the target set by some small amount and allow the system to terminate at any point in the enlarged set. This view is more realistic, since there is no reason to demand that a practical system satisfy the boundary condition exactly. To accomplish this in the random-search method, we will require

$$J < \varepsilon \tag{8}$$

where

$$\varepsilon = (\varepsilon_D, \varepsilon_V, \varepsilon_P)$$

The values of ε_D and ε_V are dictated by how closely it is desired that the system states approach the desired target set. The value of ε_P is more difficult to choose because of its lack of physical interpretation. Fortunately, the hybrid computer approach makes it easy to choose a sufficiently small value by experimentally observing the sensitivity of the solutions to small changes in ε_P. In an experimental study to be described later, the solutions were quite insensitive to variations in ε_P over a wide range.

Nonadaptive Random-Search Algorithm for Free Time

In the free-time problem one is interested in minimizing system performance at some time within an arbitrary interval of time [0,T]. (Note that we use the same symbol T that we used in the fixed-time case but the meaning is slightly changed.) There are two essential differences between the fixed-time and the free-time problems that must be accounted for in the algorithm as will now be discussed.

The first change in the algorithm required for free-time problems is the introduction of a suitable metric J^k on which to base the iterations. We will take J^k to be defined by the following equation:

$$J^k \equiv \left[\min_t J(t)\right]^k \equiv [J(t_m)]^k \qquad 0 \leq t \leq T \tag{9}$$

where t_m is the value of t at which the minimum occurs on the kth iteration. By way of comparison, in the fixed-time case the J^k used in the algorithm was taken to be $J(T)$ (i.e., the boundary-value error, computed from the state and adjoint variables at the fixed time T). In the free-time solution, however, we require $J(t)$ to be computed continuously over the interval [0,T] and its minimum value to be determined. These required changes in implementation are indicated in figure 2 by the metric computation block. The minimum operation in equation (9) is implemented via an analog circuit that "remembers" the minimum value of $J(t)$, designated in equation (9) by $J(t_m)$, on the interval [0,T] for each kth trial. Then at time T, on the kth trial, this value of $J(t_m)$ is read by the digital machine and is stored as J^k. Note that when a solution is obtained, t_m on that trial becomes the a priori unknown solution time, t_{ms}.

A second change required in the algorithm for free-time problems depends on the nature of the system to be controlled. For autonomous systems the theory requires the additional condition that the Hamiltonian $H = (p,f)$ be identically zero over the interval [0,T]. This can be done by generating $n - 1$ elements of the n-dimensional vector $p(0)$ randomly, and computing the remaining element to satisfy the requirement that $H = 0$. The required change in implementation is indicated in figure 2. The necessary computations are simple: they can always be accomplished by the solution of a

linear algebraic equation as long as we are confined to problems of mechanics. The reason is that there will always be some of the state differential equations in which the control does not appear so that the Hamiltonian will be linear in the corresponding adjoint variables. Imposing $H = 0$ at the initial time is sufficient since the theory ensures that H is constant throughout the interval.

Nonautonomous systems require a slightly different change in the algorithm. A certain constraint must be satisfied on the Hamiltonian at the final time, and this constraint could be implemented by including it as an added component of the vector metric. Since no applications of this type were considered, we will not discuss this case further.

It is now clear that with the exception of minor changes, the iterative procedure is identical for the free-time and fixed-time problems. Similarly, the adaptive aspects of the search algorithm to be discussed in the next section apply to both problems.

The Adaptive Random-Search Algorithm for Fixed and Free Time

The pure random search described in the preceding sections is generally unsatisfactory because of the excessively long convergence times, as will be seen in a later example. The convergence properties can be improved by making the system adaptive; the adaptive properties are achieved by varying the mean m and standard deviation σ as a function of the system's past performance. Since performance is determined by the input and output sequences $\{\xi^k\}$ and $\{J^k\}$, we will make the mean and standard deviation adaptive of the form

$$\left.\begin{aligned} m^{k+1} &= f_m^{k+1}(\xi^1,\xi^2, \ldots,\xi^k,J^o,J^1, \ldots,J^k) \\ \sigma^{k+1} &= f_\sigma^{k+1}(\xi^1,\xi^2, \ldots,\xi^k,J^o,J^1, \ldots,J^k) \end{aligned}\right\} \tag{10}$$

A fundamental notion in the algorithm will be that of a "success" or "failure" defined by a function of the cost J^k on the kth iteration and the smallest preceding cost J^l obtained on the last successful iteration (the lth iteration). The most frequently used definitions were

$$\left.\begin{aligned} &\text{success:} \quad J^l - J^k > 0 \\ &\text{failure:} \quad J^l - J^k \not> 0 \end{aligned}\right\} \tag{11}$$

Obviously a successful iteration is both necessary and sufficient for the system to be noninferior. Practical limitations made it necessary to modify the statement of equation (11) as follows. By definition, each component of J^l is a monotonic decreasing function of the number of iterations. Since it is possible that some components of J^l may be reduced to small values approaching the resolution of the computer while other components remain large, further reduction as defined by equation (11) would be impossible. Thus the remaining large components could not be made less than their respective ε values. This situation was observed to occur quite regularly. To

avoid it we will not require for a "success" a further reduction of those components of J^l that are already equal to or less than their respective ε values. An exact statement of this concept results in a complex Boolean statement to be described later.

Implementation of the adaptive part of the algorithm is illustrated in figure 1 by the dotted lines and the algorithm logic block. The algorithm logic block performs the computations in the above equations by utilizing: (1) the p^k information from the memory M_p, and (2) the system performance information J^k from the memory M_j. The output of the algorithm logic block is then the mean and standard deviation of the distribution for the next iteration as indicated.

The rationale behind the particular adaptive law to be used here is based on the notion of a creeping, expanding, and contracting search. The creeping character of the search is provided by varying the mean of the distribution so as to equal the initial condition of the adjoint vector on the last successful iteration. The expansion and contraction character of the search is provided by varying the standard deviation such that the search is localized when successful but gradually expanded when not successful. These characteristics give the algorithm some useful search properties.

The adaptive law for the mean value of the distribution was taken to be:

$$m^{k+1} = \begin{cases} p^k(0) & \text{if } J^k < J^l \\ m^k & \text{if } J^k \not< J^l \end{cases} \tag{12}$$

where the initial values are $m^1 = 0$, and J^o is the value based on the initial values $x(0)$. That this law is of the form of equation (10) can be seen when equations (4) and (12) are combined sequentially. Thus, the first few terms of the sequence are

$$m^1 = 0$$

$$m^2 = \begin{cases} \xi^1 & \text{if } J^1 < J^o \\ 0 & \text{if } J^1 \not< J^o \end{cases} = f_m{}^2(\xi^1, J^o, J^1)$$

$$m^3 = \begin{cases} \xi^1 + \xi^2 & \text{if } J^2 < J^1 < J^o \\ \xi^1 & \text{if } J^2 \not< J^1 < J^o \\ \xi^2 & \text{if } J^1 \not< J^o,\ J^2 < J^o \\ 0 & \text{if } J^2,\ J^1 \not< J^o \end{cases} = f_m{}^3(\xi^1, \xi^2, J^o, J^1, J^2) \tag{13}$$

The adaptive law for the standard deviation σ of the distribution was chosen so as to expand or contract the size of the search, depending on

performance. Let J^{ℓ} be the metric obtained on the last successful iteration. Then the law for the standard deviation following the unsuccessful kth iteration is

$$\sigma^{k+1} = \left\{ \begin{array}{lll} \sigma_1 & \text{if } J^k \not< J^{\ell} & \forall k: \ \ell < k \leq \ell + q \\ \sigma_2 & \text{if } J^k \not< J^{\ell} & \forall k: \ \ell + q < k \leq \ell + 2q \\ \sigma_3 & \text{if } J^k \not< J^{\ell} & \forall k: \ \ell + 2q < k \leq \ell + 3q \\ . & & . \\ . & & . \\ . & & . \\ \sigma_\gamma & \text{if } J^k \not< J^{\ell} & \forall k: \ \ell + (\gamma - 1)q < k \leq \ell + \gamma q \end{array} \right\} \tag{14}$$

where σ_i are constants such that $\sigma_1 < \sigma_2 < \ldots < \sigma_\gamma$. If, as a result of any kth trial, $J^k < J^{\ell}$, then we set the index $\ell = k$ and repeat the above variance sequence. In other words, immediately following a success, σ_1 is used for the next q iterations, σ_2 for the following q iterations, and so on, until a success is obtained. These equations are a simple form of equations (10). Occasionally, depending on the difficulty of the particular problem being studied, no success will be achieved in the γq iterations. For this reason, we repeat the search in equations (14) a number of times C (usually twice), and then reinitialize the entire search if no success is achieved. Available parameters are q, γ, and the σ_i. In the example to be discussed later, values of $q = 100$ and $\gamma = 10$ were chosen, but they do not seem critical. The choice of σ values is more important because it affects the range of the search; however, it is not difficult to choose reasonable values. A reasonable lowest value, σ_1, can be determined by observing on the display system (yet to be described) the metric J on every iteration, and choosing a value small enough that the value of J varies only slightly. For the orbit-transfer problem discussed later, $\sigma_1 = 0.1$ V proved satisfactory. The upper value is chosen to cover some sizable portion of the entire space; a value of 10 V seems reasonable for the 100 V space available on the analog computer that was used. In between, the steps are in a geometric progression enabling the search to expand rapidly. The sequence ($\sigma_1, \sigma_2, \ldots, \sigma_\gamma$) will be called the normal sequence, and the algorithm using this sequence will be referred to as the normal strategy. (A second sequence and strategy will also be required, as discussed later.)

Several additional strategies were incorporated into the algorithm to provide better convergence properties and greater versatility in certain situations. They are:

1. Single-step strategy. This strategy is intended to take advantage of favorable local properties of the surface. It is based on the idea that the step following a successful step k should not be random but deterministic in the same direction and of the same amount. That is, p^{k+1} is defined by

$$p^{k+1} = m^{k+1} + \xi^k \quad \text{if} \quad J^l - J^k > 0 \tag{15}$$

The cost in time, one iteration, is insignificant, and the benefits are substantial, as will be seen in a later example.

2. Threshold strategy. In this strategy, the requirement for a success is modified so that the difference between the cost for the kth iteration and that for the last successful iteration (i.e., the left side of eqs. (11)) must exceed some threshold value dependent on the cost function. We take the case in which the threshold is simply ηJ^l where $0 < \eta < 1$. Thus, a success is defined by

$$J^l - J^k > \eta J^l \tag{16}$$

This strategy has the desirable characteristic that smaller improvements are required as the minimum is approached.

3. Localized end-search strategy. This end-search strategy is intended to localize the search when in the immediate neighborhood of the minimum. Thus, we effect a greater resolution in searching the immediate neighborhood of the $p^l(0)$ that produced J^l when J^l itself is in the neighborhood of a solution. This is done simply by reducing the specific values of $(\sigma_1, \ldots, \sigma_\gamma)$ chosen for the normal strategy by a constant factor α, where $0 < \alpha < 1$. More precisely, when $J^l < \delta$, where δ is some small value on the order of 2ε, σ will vary over the variance sequence, but the specific values of the sequence will be a fixed fraction, α, of the values of the sequence for the normal strategy. The algorithm using this end-search sequence will be referred to as the "end-search strategy." This strategy was suggested by experimental evidence indicating a certain sensitiveness of J^l to $p(0)$ in the neighborhood of a solution; it was used sparingly, however, since it was beneficial in reducing convergence time only under certain conditions. Although a gradient method could be used in this final phase, the random-search method accomplishes the same objective without an implementation change.

4. Initial-search strategy. Since no a priori information is available that will assist in making a suitable first guess of $p(0)$, it would seem natural to assign an equal probability to all the values $p(0)$ might assume. For this reason, values of $p(0)$ are selected from a uniform distribution until a success is achieved. When this occurs, the succeeding selections of $p(0)$ are made from the gaussian distribution with mean and variance as described above. The process of selecting $p(0)$ from a uniform distribution of width $-W$ to W will be referred to as the initial-search strategy. The size of W is a compromise: It must be large enough to include possible solutions but not so large that the volume being searched is excessive. The choice of W depends, of course, on the particular problem; it is not difficult to choose a reasonable value experimentally.

It is clear that the behavior of the adaptive algorithm is different from the pure random or nonadaptive search described before. The vector metric J that measures the disparity between desired and actual terminal-boundary conditions is sequentially reduced rather than being reduced in one

iteration. The way in which this occurs is illustrated in figure 3, where a representative surface is given in only two dimensions. Because of its adaptive character, the mean of the distribution moves, after any success, to the last successful p. At this point the search starts locally and increases in a geometric progression until the next success is obtained. In this way, the successive values of J may jump from one valley to another, as indicated by the numbered points, until a J_i less than the required ε is reached. The algorithm logic block decides when this condition occurs.

The virtues of the random-search approach are clear. It has desirable local and global search properties so that "hanging up" in local valleys as with gradient methods is avoided. Further, the surfaces may be discontinuous and have many peaks and valleys. The main question is, of course, the convergence time, which is best studied experimentally with an example.

IMPLEMENTATION

The hybrid computer proved to be the most feasible way to implement the random-search algorithm. In general, there are three basic computational techniques available to the experimenter for the implementation: analog, digital, or a combination of the two, hybrid. Of prime importance in the selection of the computer system is a consideration of the number of iterations required to find a solution, and this number is not known a priori. For a second-order system, pilot studies indicate something on the order of 100 iterations to obtain a solution. Since the volume of the space increases so rapidly with the order of the system, one might expect several thousand iterations to be necessary for an increase in system order to perhaps five or six. Thus, we see that the time per iteration will be of critical importance and will largely dictate the means for implementing the search algorithm.

Each iteration can be divided into two steps: (1) integration of the equations of motion on the interval [0,T], and (2) execution of the algorithm. For the two problems to be discussed later, an IBM 7094 machine requires 1-10 sec to perform the integration in step 1. An analog computer, however, performs the same integration in 1-10 msec with an accuracy of about 5 percent relative to digital machine solutions. Thus for this specific example, the analog computer is about 1,000 times faster than the digital computer in performing the integration in step 1. The second step is best accomplished digitally. The time required to perform the second step on a digital computer of speed comparable to the IBM 7094 is the same order of magnitude as the time required for the analog to perform step 1. Therefore, a great saving in computing time can be realized over a completely digital simulation by a hybrid approach. It is worth noting that an alternative approach was investigated, utilizing pseudohybrid techniques; that is, an analog computer and limited digital logic. However, our experience shows that inaccuracies, limited storage, and limited flexibilities in logical operations seriously restrict the feasibility of this approach.

In the hybrid implementation used here, the analog computer was delegated the task of solving the state, adjoint, and control equations, as given in equations (3). It also served as the point at which the operator exercised manual control over the hybrid system. The digital computer was required to provide storage, implement the necessary logic, generate the initial conditions for the adjoint equations, and finally, to oversee the sequencing of events of the iterate cycle. One remaining task, that of calculating the metric, was performed either on the analog or digital computer, depending on the type of problem; for fixed-time problems this calculation is accomplished by the digital computer, while for free-time problems the metric must be computed by the analog computer.

A complete discussion of the implementation is presented in the appendixes. Appendix A describes the hardware; appendix B discusses the algorithm flowgraphs in detail.

BEHAVIOR AND CHARACTERISTICS OF THE ALGORITHM

This section studies the behavior of the algorithm, some of its important properties, and the effect of some of its possible variations. Naturally, the results depend on the particular problem - in this case, an orbit-transfer problem, which is described in detail in the next section.

The manner in which the system iterates and searches for a solution can be illustrated by the use of two different displays. The first display, illustrated in figure 4, gives information about the successive improved iterations. This figure shows the successive improved values of the initial adjoint vector $p(0)$ and the corresponding decrease in the boundary cost functions J_D, J_V, and J_P. It is worth noting that the $p_i(0)$ values can creep to values outside the initial square distribution. This is illustrated by the $p_3(0)$ value, which eventually crept to a solution value of 24 V even though the initial square distribution was +15 V and the maximum standard deviation of the gaussian distribution was $\sigma_\gamma = 10$ V. This situation occurred often. The second display, illustrated in figure 5, provides information about every iteration. Here the values of the boundary-cost function J_D, J_V, and J_P are shown for every iteration and also for just the successful iterations. When no improvement is obtained in the J vector, the search gradually enlarges until an improvement is found. At this point the search is localized. It is easy to select a rational value for the lower limit of the standard deviation σ_1 with this type of display.

The convergence time to obtain a solution is certainly one of the central considerations in the random-search method. Since this time is a random variable, data were taken to give suitable averages and some indication of their accuracy. In terms of the number of iterations N required for a solution, it was found from 70 solutions that $\bar{N} = 7{,}724$ and $\sigma_N = 5{,}142$. Thus, the average time required to obtain a solution is about 1.25 min. The value of σ_N gives some indication of the accuracy to be expected for measurements of other situations. For example, if 25 trials are made for another situation,

we would expect the standard deviation of the average to be $\sigma_{\overline{N}} = \sigma_N/\sqrt{25} \approx 1{,}000$. This relation is only an indication, however, since the standard deviation σ_N is not likely to remain the same for other situations.

Next, we will examine some of the possible variations in the algorithm and its parameters to indicate their relative effects on the convergence. Although these effects are studied for the given problem situation, the trends are generally valid for other situations.

First, the effect of the threshold strategy is shown in figure 6. The value of $\eta = 0$ corresponds to removing the threshold strategy. Also, a standard deviation is shown on the curve. It is worth noting that only a small improvement can be obtained by increasing η.

Second, the effect of a one-step strategy, an end-search strategy, and an increased initial search size are given below with standard deviations as indicated.

Algorithm strategies		Average number of iterations
Algorithm with	One-step strategy End-search strategy Normal initial search size	7,724 ±610
Algorithm without	One-step strategy	12,707 ±1,150
Algorithm without	End-search strategy	7,065 ±1,626
Algorithm with	Twice initial search size	9,971 ±766

The one-step strategy is fairly effective since its deletion increases convergence time by 65 percent, it was always beneficial in terms of convergence, cost very little in search time, and therefore was always used. The end-search strategy is slightly deleterious for the given situation and increases convergence time by 10 percent. This increase is not too significant, however, in view of the likely standard deviation. The end-search strategy appears to be of greatest value when the algorithm has iterated to the neighborhood of the minimum and has difficulty in meeting the boundary conditions; such a situation might exist perhaps in the neighborhood of a very sharp valley. In these cases, the end-search strategy greatly reduces the size of the search and enhances the certainty of finding the minimum. The effect of initial-search size on convergence is interesting. It is surprising to note that an increase in the initial-search size of 100 percent in all five adjoint variables (32 times larger volume) resulted in an increase of only 30 percent in convergence time.

To illustrate the effect of different J^o on the convergence time, data were obtained in which J^o was chosen to be a fraction of the value based on the initial $x(0)$ and $p(0)$. The effect of different fractional values is

shown in figure 7, where it may be seen that convergence is fairly flat within a wide range. At the lower values, the convergence time increases sharply, because the search approaches the pure random search.

AN EXAMPLE PROBLEM - ORBIT TRANSFER

In this section the usefulness of the random-search algorithm in solving a moderately difficult problem will be discussed. An orbit-transfer problem was selected because it is high order, nonlinear, and the type of problem for which optimization is appropriate; that is, since such large amounts of fuel are involved in effecting an orbit transfer, it would be profitable to minimize the required fuel. The first four subsections cover formulation of the problem, the equations that result from an application of the Maximum Principle, and the boundary conditions and the associated metric; the last three subsections present some experimental results on the characteristics of the orbit-transfer problem.

Formulation

The particular orbit-transfer problem considered was a transfer from an earth-moon trajectory to a circular orbit around the moon. The physical problem is illustrated in figure 8 for the planar case. The final desired orbit is 190 km above the lunar surface and has a radius 1928 km. The earth-moon trajectories were assumed to be hyperbolic in the vicinity of the moon; because of midcourse correction errors, the vehicle might be on any of a number of hyperbolic trajectories, which may or may not coincide at pericynthian with the desired final orbit. This situation is depicted in figure 8 by orbits 1, 2, and 3; orbit 2 is the only one that is tangent to the desired circle. The problem is to find the time histories of thrust magnitude and thrust orientation required to effect a transfer from any one of these orbits to the circular orbit in a fixed time and with a minimal use of fuel.

For purposes of simulation, the rotating coordinate system shown in figure 8 describes vehicle motion; the system origin is on the circular orbit, and its velocity is the same as for a particle in that circular orbit. This coordinate system is desirable because it subtracts large constant values from the inertial coordinate system. If the vehicle is assumed to be controlled by gimbaling a thrusting engine in which the mass flow rate is used to vary the thrust, the exact equations of motion in the rotating coordinate system shown in figure 8 are:

$$\left.\begin{aligned} m(t)\ddot{X} - 2\omega m(t)\dot{Y} &= -\dot{m}(t)c\cos\alpha + m(t)\left(\omega^2 - \frac{\mu}{r^3}\right)X + m(t)\left(\omega^2 - \frac{\mu}{r^3}\right)r_c \\ m(t)\ddot{Y} + 2\omega m(t)\dot{X} &= -\dot{m}(t)c\cos\beta + m(t)\left(\omega^2 - \frac{\mu}{r^3}\right)Y \end{aligned}\right\} \qquad (17)$$

The gimbaling implies the constraint

$$\cos^2 \alpha + \cos^2 \beta = 1 \tag{18}$$

In the above equations $m(t)$ is the vehicle mass, ω is the angular velocity of the rotating coordinate system, c is the exhaust velocity, α and β are the angles between the thrust vector and the X and Y axes, respectively, r is the radius to the vehicle in the inertial coordinate system, r_c is the radius of the desired circular orbit, and μ is a gravitational constant. Typical values used were $m(0) = 39{,}096$ kg, $c = 3.1405$ km/sec, $r_c = 1928.68$ km, $\mu = 4{,}890$ km^3/sec^2, $[\dot{m}(t)]_{max} = -31.07$ kg/sec, $\omega = 8.259 \times 10^{-4}$ r/sec.

If we let $x_1 = X$, $x_2 = \dot{X}$, $x_3 = Y$, $x_4 = \dot{Y}$, $x_5 = m(t)$, and $u_1 = \cos \alpha$, $u_2 = \cos \beta$, $u_3 = -\dot{m}(t)$, the equations of motion can be put in the state vector form

$$\dot{x} = f(x,u)$$

where $x = (x_1, x_2, x_3, x_4, x_5)$ and $u = (u_1, u_2, u_3)$. In expanded form we have the set of five nonlinear differential equations:

$$\left.\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_1 &= 2\omega x_4 + \frac{c u_1 u_3}{x_5} + \left(\omega^2 - \frac{\mu}{r^3}\right) x_1 + \left(\omega^2 - \frac{\mu}{r^3}\right) r_c \\
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= -2\omega x_2 + \frac{c u_2 u_3}{x_5} + \left(\omega^2 - \frac{\mu}{r^3}\right) x_3 \\
\dot{x}_5 &= -u_3
\end{aligned}\right\} \tag{19}$$

The terms involving r, which are difficult to simulate accurately, provide small corrections to the more dominant terms involving only the states in the rotating coordinate system. However, a simpler model, which ignores these correction terms, introduced significant errors in end states and fuel required.

The task is to minimize the fuel required to transfer from a given position on the hyperbolic orbit to the circular orbit in a fixed time T. Thus, the quantity of fuel used is introduced as the added coordinate;

$$x_o(t) = \int_o^t u_3(\tau)\, d\tau \tag{20}$$

and we can interpret the objective as the minimization of the terminal value $x_o(T)$. This additional coordinate requires the corresponding differential equation

$$\dot{x}_o(t) = u_3(t) \tag{21}$$

to be adjoined to the set (19).

Maximum Principle Solution

The solution for optimal control centers around the Hamiltonian defined by

$$H[p(t),x(t),u(t)] = (p,f) \tag{22}$$

where the p and f vectors are augmented by the auxiliary variables p_o and x_o, and (p,f) is the inner product. For the orbit-transfer problem,

$$H = p_o u_3 + p_1 x_2 + p_2\left[2\omega x_4 + \frac{cu_1u_3}{x_5} + \left(\omega^2 - \frac{\mu}{r^3}\right)(x_1 + r_c)\right]$$
$$+ p_3 x_4 + p_4\left[-2\omega x_2 + \frac{cu_2u_3}{x_5} + \left(\omega^2 - \frac{\mu}{r^3}\right)x_3\right] - p_5 u_3 \tag{23}$$

Two important sets of equations can be derived from this Hamiltonian.

The equations of the first set are the adjoint equations defined by $\dot{p}_i = -\partial H/\partial x_i$. An approximate set of adjoint equations can be shown to be

$$\left.\begin{aligned}
\dot{p}_o &= 0 \\
\dot{p}_1 &= -p_2\left(\omega^2 - \frac{\mu}{r^3}\right) \\
\dot{p}_2 &= -p_1 + 2\omega p_4 \\
\dot{p}_3 &= -p_4\left(\omega^2 - \frac{\mu}{r^3}\right) \\
\dot{p}_4 &= -p_3 - 2\omega p_2 \\
\dot{p}_5 &= \frac{cu_3(p_2u_1 + p_4u_2)}{x_5^2}
\end{aligned}\right\} \tag{24}$$

Each of the exact adjoint equations for p_1 and p_3 consists of three terms instead of the one term shown above. Equipment limitations precluded simulation of all these terms, so an approximation was required. The approximation requiring the least hardware would result from the following considerations. If the assumption is made that r is close to the desired orbit, the terms containing r in the state equations become negligibly small, and the corresponding adjoint equations then become $\dot{p}_1 = \dot{p}_3 = 0$. As an improvement over this approximation we utilize one of the three terms in the exact $\dot{p}_1$ and $\dot{p}_3$ equations; because these terms are already available in the simulation, they do not require additional equipment.

Such an approximation for the p_1 and p_3 variables can be justified by the specific way in which they affect the control function. It was estimated that any variations of the p_1 and p_3 variables (determined by the exact differential equations) would have small influence on the differential equations for p_2 and p_4. Since p_1 and p_3 couple to the control variable through the p_2 and p_4 differential equations, it can be concluded that the influence of the variations of the p_1 and p_3 variables on the control variable and associated performance would be small.

The second set of equations is for the optimal control vector that results from maximizing H over the control set. Since the control is bounded, it is not sufficient to equate $\partial H/\partial u$ to zero and then solve the resulting expression for the optimum control. A direct maximizing procedure leads to the following result:

$$\left.\begin{aligned} u_1 &= \frac{p_2}{\|P\|} \\ u_2 &= \frac{p_4}{\|P\|} \\ u_3 &= \begin{cases} M & \text{if } v = \|P\| - \dfrac{x_5}{c}(p_5 + 1) > 0 \\ 0 & \text{otherwise} \end{cases} \end{aligned}\right\} \tag{25}$$

where $\|P\| = \sqrt{p_2^2 + p_4^2}$ and M is the magnitude of the maximum thrust. It is seen that the thrust magnitude is either on or off depending on the sign of the switching function v, and the thrust angles are continuous functions of the adjoint variables. To obtain an explicit solution, it is now necessary to solve simultaneously by means of the analog computer equations (19) for x, equations (24) for p, and equations (25) for control u, subject to certain boundary conditions yet to be discussed. The auxiliary differential equations for $x_o(t)$ and $p_o(t)$ do not couple directly to the other equations; however, $x_o(t)$ must be computed to determine the fuel.

Boundary Conditions

The boundary conditions, as yet unspecified, are the heart of the random search method. The initial values for the state vector $x(0)$ are fixed at the given values, whereas the initial values of the $p(0)$ vector are chosen by the algorithm. The desired terminal conditions can be specified in a variety of ways depending on the target set chosen. On the one hand, the end points could be treated as fixed at some point such as the origin of the coordinate system. This is somewhat restrictive. On the other hand, the end points could be treated as variables by taking the target set to be the desired circular orbit. This approach would yield the best point in the target set but would require the satisfaction of complicated transversality

conditions. This investigation followed a middle course to avoid complicating the problem and selected the desired circular orbit as the target set. However, the end point was considered fixed at whatever specific point on the circular orbit (target set) is reached in the fixed time T. The advantage of this specification is that the boundary conditions are somewhat simpler than if the more complex transversality conditions had been included. With these boundary conditions, the vehicle will reach some point on the desired circular orbit but perhaps not the best point. However, as we will see in a later example, the fuels for the large majority of solutions were so close to the theoretical minimum fuel based on impulsive orbit transfer that satisfying the more complex transversality conditions could not appreciably improve the solution. This conclusion was found valid over the range for which the problem was studied. The specific values for the terminal boundary conditions can be found from conventional theory and summarized as:

$$\left.\begin{array}{ll} x_1(T), x_2(T), x_3(T), x_4(T) & \text{fixed} \\ x_5(T) & \text{free} \\ p_1(T), p_2(T), p_3(T), p_4(T) & \text{free} \\ p_5(T) = 0 & \text{fixed} \end{array}\right\} \tag{26}$$

The Vector Metric

For purposes of satisfying the boundary conditions, the components of the metric J need to be specified. Since the target set is taken to be the circular orbit, a suitable displacement metric is

$$J_D = |r(t) - r_c| \tag{27}$$

where r_c is the radius of the desired circular orbit. As for the velocity metric, it is clear that for the vehicle to stay close to the desired circular orbit after the terminal time T, the radial component of the inertial velocity V_R should be small, and the tangential component V_T should be close to the velocity $r_c\omega$ for the circular orbit. Thus, a reasonable metric that measures the errors in these velocity components is

$$J_V = \sqrt{(r_c\omega - V_T)^2 + V_R^2} \tag{28}$$

The quantities V_T and V_R are simple transformations of the states in the rotating coordinate system. The only adjoint variable that must be fixed at the terminal time is p_5, so that we use simply

$$J_P = p_5^2(T) \tag{29}$$

The components of J must be reduced to values that depend somewhat on the mission and accuracy requirements after the terminal time. These values were chosen to be

$$\left.\begin{aligned} \varepsilon_D &= 8 \text{ km} \\ \varepsilon_V &= 15 \text{ m/s} \\ \varepsilon_P &= 0.5 \text{ V} \end{aligned}\right\} \tag{30}$$

The velocity requirement, ε_V, reduces the terminal inertial velocity to less than 1 percent of the initial value. The adjoint variable requirement was readily found experimentally. As mentioned in the discussion following equation (8), values of ε_P several times this value were found to be still satisfactory.

Results for Orbit-Transfer Problem

In the particular orbit-transfer situation discussed in this section, the vehicle is approaching the moon on hyperbolic orbit number 2 (fig. 8). Pericynthian coincides with the desired final circular orbit, which is 190 km above the lunar surface. The initial starting point was chosen as -37°, and the time allowed for the vehicle maneuver was taken to be 600 sec. For comparison, the time to reach pericynthian with zero control is 534 sec.

A typical solution obtained by the random-search algorithm is illustrated in figure 9 by the complete set of time histories; shown are the system states in the rotating and inertial coordinate systems, the control vector, and the adjoint variables. Perhaps the most interesting are the velocities x_2 and x_4 in the rotating frame, which are reduced from large values of around -900 m/sec and 400 m/sec to small values so that the inertial velocity components V_R and V_T can be reduced to near their desired values. Also the quantity $r - r_c$, the difference between the vehicle terminal radius and the desired circular orbit radius, is reduced to a small value so that the vehicle will arrive near the desired orbit. The terminal inertial value $\mathscr{Y}$ is slightly positive, indicating the desired orbit was attained slightly past pericynthian. The optimal control vector is also of interest; it is indicated that the thrust should be turned on at about 260 sec before the fixed final time and directed as shown in figure 9(e).

Average performance and its variation are perhaps more significant than an individual solution. The time histories of 70 solutions varied only slightly. In terms of the end states and fuel used, the following statistics were obtained:

$$\begin{aligned} \text{Av } \mathscr{X} &= 1{,}935 \pm 5 \text{ km} \\ \text{Av } \mathscr{Y} &= 33 \pm 8 \text{ km} \\ \text{Av}[r(T) - r_c] &= 6 \pm 2 \text{ km} \\ \text{Av}(V_T - r_c\omega) &= 2 \pm 5 \text{ m/sec} \\ \text{Av } V_R &= 1 \pm 12 \text{ m/sec} \\ \text{Av fuel} &= 10{,}230 \pm 80 \text{ kg} \end{aligned}$$

It is worth noting from the data for $\mathscr{X}$,$\mathscr{Y}$, and $r - r_c$ that the set of reachable points is centered slightly farther out than the desired circular orbit and slightly past pericynthian; the standard deviation shows this set is confined to a very small region. The inertial velocities have been substantially reduced from the initial total magnitude of approximately 2400 m/sec. The fuel data are interesting because the figures are so close to the idealized minimum-fuel impulsive orbit transfer of 10,120 kg.

It is significant to note that from this same set of 70 solutions, the solutions for the initial condition adjoint vector p(0) were vastly different even though the time histories of the system states were so close. The extremes observed as well as the p(0) vector corresponding to figure 9 are as follows:

Adjoint variable	Extremes	Values from figure 9
p_1	-31.98 to +7.15	-11.36
p_2	+1.08 to +14.67	+8.96
p_3	-8.33 to +35.60	-.54
p_4	-11.94 to +5.45	-9.25
p_5	-7.98 to +8.11	-6.30

These results are significant because they illuminate the nature of the mapping $p(0) \rightarrow J$. In geometrical terms, discussed in more detail later, these results mean that the boundary cost-function hypersurfaces have many stalactites protruding below the ε level.

Variations in the time allowed to reach the desired orbit were studied with much the same type of results described in the preceding paragraphs. Solutions were obtained for times from 550 to 850 sec (the time to reach pericynthian with zero control was 534 sec). The main difference in results obtained was that as the allowable time increases, the set of attainable points as given by $\mathscr{X}$ and $\mathscr{Y}$ moves farther around the desired circular orbit and the fuel requirements appear to increase. For example, for T = 850 sec, $\mathscr{X} \approx$ 1828 km, $\mathscr{Y} \approx$ 658 km, and the fuel required increased by about 300 kg. When the allowable time was outside the range given above, no solutions were obtained.

Results for Other Orbit-Transfer Situations

The algorithm was experimentally tested on a variety of physical situations, which include different initial points along each of the three trajectories as indicated by the grid of points in figure 8, different allowable times for the vehicle to reach the desired point, and variations in some of the vehicle design parameters.

Figures 10 and 11 show two different solutions to the same problem in which the starting point is farther along on orbit 2 (-13°) than in the previous two sections. These solutions are interesting because of the unusual control functions found. One of the control solutions has two thrusting periods (on-off-on) with the initial thrusting being quite long; the other solution has only one thrusting period (on-off) and again the initial thrusting is long. In contrast, most of the solutions found for other situations have only a single thrusting period of the off-on type.

Figures 12 and 13 illustrate solutions obtained on orbits 1 and 3. On orbit 1, the solutions showed there was always one thrusting period of the off-on type (see fig. 12) for all starting points along the orbit and all times T for which solutions could be obtained; the fuels used were close to the values obtained on orbit 2. On orbit 3, the solutions nearly always consisted of the more unusual two thrusting periods of the on-off-on type (see fig. 13). It was also noted that solutions were quite difficult to obtain on orbit 3 and that the fuels required were approximately 1500 kg greater than on orbits 1 and 2.

It was also demonstrated that the random-search approach could be of considerable value in preliminary vehicle and/or mission design studies. For this purpose, the effects of varying the initial vehicle design parameters, thrust level and mass, were investigated. It is significant to note that solutions could readily be obtained with thrusters of 75 and 50 percent of the normal thrust capability used in the previous portions of the study. Similarly, there were no difficulties in obtaining solutions for different initial vehicle masses.

Boundary Cost-Function Surfaces

Perhaps the most illuminating results were revealed by an experimental study of the mapping: $p(0) \rightarrow J$. Geometrically, this mapping can be interpreted as three hypersurfaces (representing the components of the metric J), which are functions of the five adjoint initial conditions. The character of the cross sections through these surfaces at a solution point is given in figures 14 and 15 for the two different problem situations indicated. These curves were obtained by slowly varying one of the variables from -100 V to +100 V while all the others were fixed at their respective solution values. In this manner, cross sections through the hypersurfaces passing through a solution point were plotted. Note the dip in all three surfaces at the optimum value of the adjoint variable. The interest in these curves is, of course, in the rather irregular, multivalleyed nature of the surfaces as well as the rather narrow valleys surrounding the optimum point. The sharpness or narrowness of this valley gives an indication of the difficulty in finding the solution. Further, these surfaces clearly indicate the difficulties that gradient methods would have because of the likely possibility of "hanging up" in the wrong valley. The flat regions in figures 14 and 15 are actually plateaus in the parameter space and not saturation of the computer equipment, with the exception of the flat regions at the extreme left and right sides of the J_p curves. Such plateaus occur because for certain ranges of the $p(0)$

parameters the switching function is such that the control u_3 is zero. These plateaus would present problems to gradient procedures. The two-dimensional cross sections shown give only a hint of the difficulties to be encountered in the actual six-dimensional space.

Another more pictorial representation for the problem situation in figure 14 is given by the three-dimensional views in figure 16. Here are shown the boundary cost-function surfaces in the p_1,p_2 plane and in the p_1,p_4 plane, while all other adjoint variables are fixed at their solution values. These results illustrate the character of the surfaces away from the optimum point in two directions instead of only one direction.

Further clues as to the nature of the boundary cost-function surfaces are revealed by previously presented data, which showed that many distinctly different $p(0)$ vectors result in solutions and that the surfaces have many stalactites protruding below the ε level.

AN EXAMPLE PROBLEM - SATELLITE ATTITUDE ACQUISITION

In this section, the random-search algorithm is applied to the single-axis attitude acquisition control problem, where the time to acquire the desired boundary conditions is left free. The problem is formulated through a physical description of the problem and derivation of the exact equations of motion. Second, the equations necessary for determining optimal nonlinear control as derived by means of the Maximum Principle are outlined, as well as, for comparative purposes, the optimal proportional control derived in reference 14. Next, the boundary conditions and the vector metric are discussed. The final two subsections illustrate the computer results through a variety of time history solutions and fuel performances, and some cross sections through the boundary cost-function hypersurfaces.

Formulation

Consider a rigid body V rotating about its center of mass and fixed in inertial space. A set of axes will be taken fixed to the vehicle and alined with the principal axis of inertia with its origin at b_o, the center of mass (fig. 17). Finally, consider the vehicle to be inertially asymmetric and to be in a general tumbling motion at some arbitrary initial time.

The objective of the control is to apply torques in a manner that will reduce total momentum to zero and orient a particular axis of the vehicle from any initial orientation to any other prescribed orientation in inertial space. The control torques required to orient the vehicle are assumed to be produced by mass expulsion devices alined in each of the three body axes. It is assumed that the torques produced by these devices are proportional to mass flow rate and that the flow rates are bounded at some maximum value (except when proportional control is examined). The vehicle is to be controlled with the least amount of fuel and in any time within an arbitrary interval $[0,T]$.

The dynamic equations of motion of a rigid body rotating about a point fixed in inertial space and acted on by external torques are given by the following set of equations:

$$\left.\begin{aligned} \dot{\omega}_1 &= \frac{1 - \beta}{\alpha}\,\omega_2\omega_3 + \frac{\gamma_1}{\alpha} \\ \dot{\omega}_2 &= (\beta - \alpha)\omega_3\omega_1 + \gamma_2 \\ \dot{\omega}_3 &= \frac{\alpha - 1}{\beta}\,\omega_1\omega_2 + \frac{\gamma_3}{\beta} \end{aligned}\right\} \tag{31}$$

where

$\alpha = I_1/I_2$ — Roll-to-pitch inertia ratio

$\beta = I_3/I_2$ — Yaw-to-pitch inertia ratio

$\gamma_1 = M_1/I_2$, $\gamma_2 = M_2/I_2$, $\gamma_3 = M_3/I_2$ — Control accelerations (rad/sec^2) normalized to I_2

ω_i = body rates, rad/sec; M_i = torque; $i = 1, 2, 3$

The kinematic equations necessary to relate the body axes to the inertial axes can be chosen in various ways. We will choose the independent set of direction cosines a_{ij} to relate the three body axes $i = 1, 2, 3$ to the jth inertial axes where a_{ij} is the cosine of the plane angle between b_i and S_j. In this study we are interested in single-axis orientation, and we will choose to orient the b_3 body axis with the inertial axis S_3 (see fig. 17). Thus, the three direction cosines (a_{13}, a_{23}, a_{33}) will be of interest. These three kinematic variables are related to the dynamic variables by the following set of differential equations (see ref. 14 for a detailed discussion):

$$\left.\begin{aligned} \dot{a}_{13} &= \omega_3 a_{23} - \omega_2 a_{33} \\ \dot{a}_{23} &= \omega_1 a_{33} - \omega_3 a_{13} \\ \dot{a}_{33} &= \omega_2 a_{13} - \omega_1 a_{23} \end{aligned}\right\} \tag{32}$$

For this study, the specific values of the roll and pitch inertia ratios were taken to be $\alpha = 1.15$ and $\beta = 0.48$. Also, the maximum control torque accelerations permitted in the nonlinear controller situation are:

$$\gamma_{1_{max}} = \gamma_{2_{max}} = 2\gamma_{3_{max}} = 7.5 \text{ m rad/sec}^2$$

To transfer equations (31) and (32) into state variable form, the following substitutions are made:

$$a_{13} = x_1 \qquad \omega_1 = x_4 \qquad u_1 = \frac{\gamma_1}{\alpha}$$

$$a_{23} = x_2 \qquad \omega_2 = x_5 \qquad u_2 = \gamma_2$$

$$a_{33} = x_3 \qquad \omega_3 = x_6 \qquad u_3 = \frac{\gamma_3}{\beta}$$

This yields the following set of state variable equations:

$$\left.\begin{aligned}
\dot{x}_1 &= x_6x_2 - x_5x_3 \\
\dot{x}_2 &= x_4x_3 - x_6x_1 \\
\dot{x}_3 &= x_5x_1 - x_4x_2 \\
\dot{x}_4 &= \frac{1 - \beta}{\alpha} x_5x_6 + u_1 \\
\dot{x}_5 &= (\beta - \alpha)x_6x_4 + u_2 \\
\dot{x}_6 &= \frac{\alpha - 1}{\beta} x_4x_5 + u_3
\end{aligned}\right\} \tag{33}$$

The objective of the control is to take the state vector from a fixed initial value $x(0)$ to a fixed final value $x_f(t_{ms})$ in an unspecified time t_{ms} within the interval $[0,T]$, while using the least possible fuel. A new coordinate, proportional to the total fuel used in all three axes, can be defined as follows:

$$x_o(t) = \int_o^t \sum_{i=1}^{3} |u_i(\tau)| d\tau \tag{34}$$

and we can then interpret the objective as the minimization of the terminal value $x_o(t_{ms})$.

Maximum Principle Solution

The nonlinear optimal control can be derived by an application of the Maximum Principle. The equations necessary for computer implementation are summarized below. First are the adjoint equations:

$$\left.\begin{aligned}
\dot{p}_1 &= p_2x_6 - p_3x_5 \\
\dot{p}_2 &= p_3x_4 - p_1x_6 \\
\dot{p}_3 &= p_1x_5 - p_2x_4 \\
\dot{p}_4 &= p_3x_2 - p_2x_3 - p_5x_6(\beta - \alpha) - p_6x_5\,\frac{\alpha - 1}{\beta} \\
\dot{p}_5 &= p_1x_3 - p_3x_1 - p_4x_6\,\frac{1 - \beta}{\alpha} - p_6x_4\,\frac{\alpha - 1}{\beta} \\
\dot{p}_6 &= p_2x_1 - p_1x_2 - p_4x_5\,\frac{1 - \beta}{\alpha} - p_5x_4(\beta - \alpha)
\end{aligned}\right\} \qquad (35)$$

Second are the equations defining the optimal control vector at each instant of time:

$$\left.\begin{aligned}
u_i(t) &= N_i \operatorname{sgn} p_{i+3}(t) \quad \text{if} \quad |p_{i+3}(t)| > 1 \\
u_i(t) &= 0 \quad \text{if} \quad |p_{i+3}(t)| < 1
\end{aligned}\right\} \qquad (36)$$

where $i = 1, 2, 3$ and N_i is the maximum torque acceleration allowed in the ith control axis. Note that the same direct method for maximizing H over the control set applies to this problem as it did for the orbit-transfer problem of the previous section. The control torque is of the on-off type, and torque direction is obtained by assigning the correct sign to the "on" signal according to equation (36).

Proportional Control Law

The optimal proportional control law used in this paper for comparative purposes was taken from reference 14, in which the structure of the law is assumed to be of the form:

$$\left.\begin{aligned}
u_1 &= Dx_4 - Ex_2 \\
u_2 &= Fx_5 + Gx_1 \\
u_3 &= Hx_6
\end{aligned}\right\} \qquad (37)$$

The parameters D, E, F, G, and H were left free, and by means of a random parameter search, suitable values were found for which the stated objective

of the problem (zero momentum and alinement of b_3 to S_3) was achieved. The search was repeated a number of times, and each time, system performance given by equation (34) was observed. An optimum parameter vector was selected from the set of parameter vectors that satisfied the problem objective and minimized the performance criteria. This parameter vector, in conjunction with equation (37), defines optimal proportional control. This control may be difficult to achieve in practice; however, since no bounds have been imposed on the thrust. When there are bounds, the control law is then referred to as optimal saturating proportional control.

The Vector Metric

The desired boundary condition at the terminal time was chosen to be zero momentum and alinement of the body axis b_3 with the inertial axis S_3; this is expressed by $x_f(t_{ms}) = (0,0,1,0,0,0)$. In the random-search approach, it is necessary to introduce the vector metric J to satisfy these boundary conditions. Its general form was specified previously to be $J = (J_D, J_V, J_P)$ where the subscripts on the components identify them as displacement, velocity, and adjoint (or transversality condition) errors, respectively. In this application, since all terminal states $x_f(t_{ms})$ are fixed, $p(t_{ms})$ is completely free so that we may ignore the J_P component in the vector metric. For the present example, J_D and J_V are taken to be

$$\left.\begin{aligned} J_D &= \sqrt{x_1^2 + x_2^2 + (x_3 - 1)^2} \\ J_V &= \sqrt{x_4^2 + x_5^2 + x_6^2} \end{aligned}\right\} \tag{38}$$

It is clear that $J_V = 0$ implies $x_4, x_5, x_6 = 0$, which represents zero momentum as desired. Also, $J_D = 0$ implies the desired final orientation $x_1 = x_2 = 0$ and $x_3 = 1$. In practice, we require only

$$(J_D, J_V) < (\varepsilon_D, \varepsilon_V)$$

where the ε values are chosen to meet the problem requirements. For the specific problem discussed below, the chosen value of ε_D required that the b_3 body axis be oriented to within 8° of the S_3 inertial axis. The ε_V value chosen represented a 0.75°/sec error rate in each body axis at the final time t_{ms}. These values are primarily determined by the accuracy with which the analog computer can calculate equation (38) over the full range of state variables between initial and final times. The above values could be made smaller if desired by an automatic rescaling or perhaps another choice of the form of the metric. This was not done here because such a reduction of the ε values would not appreciably affect the fuel consumption.

Results for Satellite Acquisition Problem

This section covers some computer solutions for the satellite attitude acquisition problem, with an emphasis on results for the optimal nonlinear

control obtained by implementing the random-search algorithm discussed in the preceding sections. For comparison, we will also give results for the proportional control studied in reference 14.

Table 1 summarizes the control systems to be studied, the initial conditions of the vehicle, and the resulting fuel requirements. In the initial condition vector, the first three components are the angular position variables given in terms of direction cosines, and the last three components are the angular velocities given in degrees per second. The desired final condition vector is taken to be $x_f(t_{ms}) = (0,0,1,0,0,0)$, which reflects the objective of control to reduce the initial momentum to zero and to aline the body axis b_3 with the inertial axis S_3.

No control.- Figure 18 shows the time histories of the state variables describing the motion of the system when no control is used and the vehicle starts with the initial condition $x(0) = (0,0,1,-10,10,10)$. This corresponds to initial alinement of axes but with the initial angular velocities indicated. The angular positions vary over their entire range (-1,1) during the time interval $[0,t_{ms}]$. These results suggest that with limited torque control, the wide range of angular position variations would persist over a substantial portion of the interval. In this event, the equations of motion are definitely nonlinear and it is inappropriate to linearize them. The Maximum Principle allows these nonlinearities to be dealt with directly.

Optimal proportional control.- The time history solutions for the optimal proportional control derived in reference 14 (as discussed above) are given in figure 19 for the same initial condition as with no control (see table 1). As is well known, optimizing system performance under the assumption of proportional control often leads to impulsive-type control. This tendency can be seen in figure 19(c) where the initial torque acceleration rises to 44 millirad/sec^2 and then decreases relatively quickly to zero. A normalized fuel consumption as measured by the computer was $x_o(t_{ms}) = 0.65$, which is nearly two and one-half times the minimum theoretical value of 0.28 obtained by allowing an ideal impulse of torque.

It should be noted that the fuel requirement of 0.65 was attained with ideal proportional control, that is, without saturation of the torque. Reference 14 demonstrated that the effect of saturation on the proportional controller is to increase both the consumed fuel and the time required to accomplish the mission.

Optimal nonlinear control.- The solutions for the optimal nonlinear control obtained by implementing the random-search algorithm are given in figures 20 and 21. These figures correspond, respectively, to the cases indicated in table 1: initial alinement and initial nonalinement.

The results given in figure 20 are for initial alinement of axes in which the initial condition vector is $x(0) = (0,0,1,-10,10,10)$, the same as for the other controllers discussed above. A comparison of figure 20 with figures 18 and 19 shows that the solutions obtained are quite different from the previous cases. The maximum torque chosen for optimal nonlinear control

was approximately one-sixth the peak value for proportional control; note the different origins for the three control functions. From careful examination of the time histories, it appears that the optimal control law is anticipatory: it acts at the most advantageous moment to reduce the large excursions of the states to the desired end values. Fuel consumption was found to be $x_0(t_{ms}) = 0.31$, which is only 1.1 times larger than for the optimal impulsive case. By comparison, the optimal proportional control is very inefficient, requiring 2.1 times the fuel for optimal nonlinear control. This factor would be even larger if the comparison were made to the more practical saturating proportional control system. The operate times in figure 20 are nearly the same as for the proportional control system; thus, we need not necessarily expect a time penalty for the practical constraint imposed by control saturation.

In the results given in figure 21, the initial condition vector $x(0)$ is not alined. The initial value is $x(0) = (0,1/\sqrt{2},1/\sqrt{2},-10,10,10)$ and corresponds to an initial rotation of b_3 of 45° in the S_2,S_3 plane. The time histories and optimal control thrust are seen to be similar to those with initial alinement. As for the fuel, we might expect that the initial angular position error would require a higher fuel consumption to achieve the same objectives. Indeed, experimental data show that fuel required is 0.43 which is 40 percent more than when the axes are initially alined.

Boundary Cost-Function Surfaces

It was pointed out previously that irregularity in the boundary cost-function hypersurfaces reflects the need for a search algorithm that incorporates both global and local search properties. The highly irregular nature of these hypersurfaces for the attitude control problem is demonstrated in figure 22, which shows typical two-dimensional cross sections through the hypersurfaces at a solution point. The curves were obtained in the same manner as described for the orbit-transfer problem. Note in particular the irregular multipeak nature of the surfaces and the occasional noisy appearance.

CONCLUDING REMARKS

A feasible computational method for solving the two-point boundary-value problem associated with the Maximum Principle has been established. The method generates explicit optimal solutions for complex, nonlinear control problems. As illustrated in this paper, the method can be used to solve problems for which optimum solutions have not been previously obtained. One of the problems solved indicates the striking improvement in performance that can be obtained by the use of optimal nonlinear control.

Although the solutions obtained are open-loop solutions, the usefulness of the technique, from the control point of view, is in its value as an off-line preliminary system design tool permitting study of trade offs among

various initial design choices in terms of performance. Furthermore, it is generally accepted that knowledge of optimum open-loop control data could be valuable in building closed-loop (on-line) controllers.

The feasibility of the method is made possible by the uniqe computational capabilities of the hybrid computer. A major advantage of such a computing system is the speed with which nonlinear discontinuous differential equations can be integrated. An approximate 1000:1 speed advantage with respect to the digital computer was obtained.

The method could be used more effectively in an on-line or off-line implementation with an increase in the iteration rate, which generally must be achieved through advances in hybrid systems, although a moderate rate increase with the present system could be realized by careful redesign of the program. Undoubtedly the most important limitation of the method is due to the minimum resolution of the analog computer. Thus, certain problems with large dynamic range might be excluded, or perhaps require complicated scale changing.

Finally it should be reiterated that the random-search algorithm could be used in a great many situations other than the two-point boundary-value problem discussed in this report. For example, some curve-fitting problems can be framed as a multiparameter search problem suitable for direct application of search techniques outlined in this paper.

Ames Research Center
National Aeronautics and Space Administration
Moffett Field, Calif., 94035, July 28, 1969

APPENDIX A

THE HYBRID-SYSTEM HARDWARE

This appendix describes the computer hardware utilized to implement the algorithm. The description will provide the reader with a basis for evaluating the adequacy of the computer equipment available to him.

Figure 23 is a hardware diagram of the hybrid system, showing the two basic elements of the simulation, the analog and digital computers along with their coupling system, and peripherals. The coupling system comprises two distinct parts: (a) the linkage system, and (b) the control interface system.

Digital Computer

The digital computer used in the optimization program was an Electronic Associates, Inc. (EAI) model 8400. It is a medium-sized, high-speed computer designed to operate in a hybrid environment. The particular machine used has 24,576 words of core memory with 32 bits per word. Memory cycle time is 2 μsec. The machine uses parallel operation for maximum speed. Floating-point operations are hardware implemented. A MACRO ASSEMBLY programming language is included to minimize the execution time. The instruction repertoire includes special commands by which discrete signals can be sent to or received from the external world. External interrupts are provided that can trap the computer to a specific cell in memory. In an example to be discussed later the optimization program utilized about 8,000 words of storage; about 1,000 constitute the actual optimization execution program, and the remaining 7,000 are used for subroutines, monitor and on-line debugging, and program modification routines.

The peripherals of the digital computer include magnetic tapes and a card reader for program input and storage, and a printer for data logging.

Analog Computer

The analog computer for the fixed-time solution was an Electronic Associates 231R-V; for the free-time solution, an Electronic Associates model 8812 was used. Both computers are of the general-purpose type and are quite conventional.

The analog computer serves as the point at which mode control of the hybrid computer is accomplished. By manual selection of switches either of two modes can be commanded:

1. In the "search" mode the analog computer operates in a high-speed repetitive manner. Such operation is accomplished by controlling the mode of the individual integrators with an appropriate discrete signal. This

signal is a two-level signal, generated on the control interface in conjunction with the digital computer, that, depending on the level, holds an integrator in either "operate" or "initial condition" mode.

2. In the "reset" mode, the integrators are placed in their initial-condition mode and held there.

Particular analog equipment worth noting are the track-store units, D/A switches, and comparators with which the control logic was implemented. At the end of the operate period T, the digital computer reads a number of variables to be used in the algorithm. Because of the high repetitive speeds, the value of a variable could change considerably between the end time T and a later time when it is actually converted. Thus, track-store units were used to hold the variables at their respective values at time T until the digital computer read all of the values. The control logic requires on-off type switching, and the high-speed electronic comparators and electronic switches were necessary for proper operation. Nonlinear operations such as multiply and divide provided no particular difficulties, although a square root operation did require the use of a diode function generator in one of the examples.

For continuous output, a display console was connected to the analog computer to provide visual readout of variables. The display contained a cathode ray tube (CRT) that could simultaneously display up to four channels, and enabled photographic records to be taken of the display quantities. The display was extremely helpful in determining if the algorithm was functioning properly. Other continuous analog data useful for determining proper functioning were $p^{\mathcal{L}}$ and $J^{\mathcal{L}}$, which could be recorded on a pen recorder since their rate of change was low.

Control Interface

The control interface between the analog and digital computers was an Electronic Associates DOS 350 for the case of the fixed-time implementation. Free-time implementation utilized the interface (as well as logic) that is a part of the Electronic Associates 8812 analog computer. It is through the control interface that the iteration process is controlled. An important task allocated to this subsystem is the operate-time control. This function is implemented through the use of a counter and is the key element in the control of all timing in the hybrid simulation. The counter is driven from a high-frequency source in the interface system allowing for a very high degree of resolution in the simulated operate time. The counter is constructed by patching modular blocks that can be combined to give a wide range of simulated operate times. Specific times within this range are selected with thumbwheel switches. Two other functions of the interface are: (1) to sense any conditions in the analog computer that can be represented by discrete variables (binary levels), and (2) to send discrete signals to the analog system to be used as control levels or indicators. An example of (1) would be the hybrid system mode control, which merely amounts to the operator depressing the "reset" or "search" switch on the analog console. This action sets a binary level that is then sensed by the digital computer. An example

of (2) is when the digital sends the operate command to the operate-time counter. The interface system allows patching of Boolean functions. Hence, some of the logic operations required for timing pulses, event signals, and other like operations were very effectively programmed here.

Linkage System

The linkage system shown in figure 23 houses the conversion equipment - the A/D (analog to digital) and D/A (digital to analog) converters. It is through here that all of the data passes between the analog and digital portions of the simulation. The linkage system is controlled by command from the digital computer.

Input to the digital computer is through the A/D converter which has preceding it a channel selection device or multiplexer to select the analog channel which is to be converted. Conversions were done sequentially through the analog channels at a maximum rate of 80,000 samples per sec from channel to channel. Conversion of 10 channels thus takes 125 μsec.

Output to the analog used the D/A converters with each data channel having its own conversion unit. The maximum conversion rate of the D/A's used is 250,000 conversions per sec.

APPENDIX B

THE HYBRID-SYSTEM PROGRAMS

This appendix covers the use of the hardware systems for implementation of the random-search algorithm. The first section discusses the sequence of events in the iterative cycle of the algorithm; subsequent sections detail the digital and analog programs.

Sequencing of Events During One Iteration

Figure 24 shows the sequencing of events during one iteration cycle; the instants of time $t_1, t_2, \ldots, t_5$ are fixed relative to each other, and the cycle begins at t_1 with the analog integrators in an operate mode. The elapsed time $(t_2 - t_1)$ is controlled by a counter on the interface system. At t_2 an interrupt pulse generated on the control interface (1) signals the digital computer to commence its operations, and (2) simultaneously instructs the track-store units of the analog computer to hold their respective values. During the interval $(t_3 - t_2)$ the digital computer reads these analog variables with the A/D converter. At t_3 the digital sends a pulse via the interface to the analog console which commands the integrators to an initial condition mode. At t_4, when the data required by the analog have been generated by the algorithm, the D/A converters send these values to the appropriate points in the analog portion of the simulation. The digital machine allows enough time for the transients to settle in the initial condition circuits of the analog before sending a command at t_5 (the same event as t_1) that places the integrators in an operate mode and starts the counter. Since t_5 and t_1 are the same event, repetitive operation is under way.

The total iterate time $(t_5 - t_1)$ is composed of two main parts: $(t_2 - t_1)$, which in the example problems was scaled in the simulation to between 2 and 10 msec; and $(t_5 - t_2)$, which was determined primarily by the speed of the digital machine in computing, converting, and generating random numbers. The second period was on the order of 8 msec. Thus, the total iterate time for the above situation is on the order of 10 to 20 msec (or 100-50 iterations per sec). This figure is dependent on the control problem chosen and the exact form of the algorithm implemented.

Digital Flowgraph, Fixed Time

Figure 25 is a program flowgraph showing the operation of the algorithm and the iteration process for the fixed-time problem. Note the inclusion of the event times $t_1, t_2, \ldots, t_5$ discussed in connection with figure 24. The program is continuously recycling in a high-speed repetitive fashion.

There are three basic loops which correspond to the three system modes in the optimization program: a reset loop, a search loop, and an end-state

loop. The reset loop is for the purpose of initializing the program. The search loop of the program uses the algorithm to search for a solution to the problem. The end-state loop is entered by the digital program when a solution is found, and is used for the generation of graphic displays. The operator manually selects the search or reset mode as discussed in the section dealing with the analog computer.

Reset loop.- The reset loop of the repetitive operation cycle initializes the program and prepares it for the search mode. When the system is in the reset mode, the integrators of the analog computer remain in their initial condition state. The flow of the reset cycle is shown in figure 26, where that particular loop is emphasized by line weight. It is while the system is in the reset mode that the digital program is first entered (at START in the figure). The first operation performed is that of initialization. It is during this process that all the program variables are set to their initial states. Also, the line printer is initialized and the data header printed. The interrupt line then is activated, the analog computer is signaled that an iteration cycle is to begin, and the counter controlling the operate time of the analog is started. The digital computer then halts and waits for an interrupt to signal that the period T has ended.

When the interrupt occurs, the digital program proceeds to the next operation where the values of the states $x^k(T)$ and $p^k(T)$ are read through the A/D converter. Once the inputs to the digital computer have been read, a pulse is sent to the analog computer to place the integrators in their initial condition mode. Since the integrators are in the reset mode, and hence already in their initial condition state, this pulse has no effect; it is needed later, however, when the system is in the search mode or the end-state mode where the integrators are not so constrained.

The next operation performed is that of calculating the metric J^k based on the samples of the state obtained in the last block. Since the terminal states are the same as the initial states in the reset mode, J^k is a function of $x(0)$ and $p(0)$. The value is thus peculiar to the reset mode and is designated J^o. One could, of course, choose an arbitrary fraction of J^o, or even an arbitrary value. Once the metric is calculated, the digital computer tests the system mode and branches to that portion of the program that is exclusive to the reset loop.

The program now initializes the algorithm by setting $m^k = p^\ell(0) = 0$ and $J^\ell = J^o$. The values of the components of $p^k(0)$ are then generated using a uniformly distributed noise source, the space of the noise being $[-W,W]$. Note that in reset mode, $p^{k+1}(0) = \xi^{k+1}$. To discuss the next operation it is necessary to consider what happens elsewhere in the program. During the search mode, the program can modify the values of the variance sequence $(\sigma_1, \ldots, \sigma_\gamma)$; that is, when certain conditions are met the values of σ_1 through σ_γ are changed to another set of values. (See discussion, p. 10, on end-search strategy.) Now, since it is possible the search strategy will have to be reinitialized (see discussion following eq. (14)), it will be necessary to restore the initial values of the variance sequence to those determined by the normal strategy. This reinitializing is accomplished in

the second to last block in this path of the flowgraph. The final step is to set σ equal to the first element of the variance sequence.

At this point the reset loop returns to the mainstream of the program by entering the output portion of a cycle. It is here that all the quantities required at the analog computer are converted to analog form by the D/A converters. The data sent to the analog include $p^k(0), p^l(0), J^l$, and J^k. Note that $p^k(0)$ is required by the analog computer program, whereas the others are simply displayed to determine if the algorithm is functioning properly.

Finally the digital program activates the interrupt line and signals the analog computer and control interface to begin another cycle. As before, the program now goes into a halted state waiting for the interval T to pass. Cycling in the reset loop continues until the operator is ready to begin a search cycle and does so by putting the hybrid system in the search mode.

The search loop.- It is in the search mode that the algorithm seeks an optimal solution to the control problem implemented. Search mode is selected by the operator at any time and is considered to begin on the next iteration.

During the first iteration of search mode, the analog computer solves the system, adjoint and control equations using the $p(0)$ established in the last cycle through the reset loop. The digital computer then receives the time T interrupt signal and reads the states $x^k(T)$ and $p^k(T)$ as shown in figure 27. Once read, the integrators are set back to the initial condition where they will await the next $p(0)$ generated by the algorithm. The next operation computes the metric as was done in the reset loop, but this time the J^k is based on the actual solution of the system equations at time T.

The system mode test is next and results in a branch to that portion of the program that generates the $p(0)$ based on the adaptive principles discussed previously. The search begins with a test to determine if the system is in end-state. Since this is the first time through the search loop, end-state condition cannot have been established so the program proceeds to the operation that tests the metric. The purpose of the metric test is to determine success or failure. However, the conditions of success or failure are more general than those indicated by equation (11); in the discussion following that equation, it was pointed out that the precise statement of the complete set of conditions determining success or failure was a complex Boolean statement. An equivalent flowgraph of this statement as it was implemented in the program is shown in figure 28, where η is the threshold factor discussed in equation (16).

Now at this point in the discussion we shall assume that J^k has been tested according to the flowgraph of figure 28. Furthermore, we will take the results of the metric test to be a success so that portion of the search loop can be examined. The first operation updates the memory of the algorithm by setting $p^l(0)$, the last successful value of $p(0)$ to $p^k(0)$, the vector which gave this success, and similarly by setting the last successful metric J^l to J^k. Next a test is made on the metric to see if it meets the requirement of a solution, that is, $J^l < \varepsilon$. Assume for the moment that it does not

and that the program proceeds out the lower branch of the test. The program then generates the next try for the adjoint initial conditions on the basis of the single-step strategy. Following the establishment of $p^{k+1}(0)$ the program tests J^{ℓ} to determine if the end-search strategy should be employed by comparing J^{ℓ} with δ. If J^{ℓ} is less, the values of the variance sequence $(\sigma_1, \ldots, \sigma_\gamma)$ are set for end-search strategy (see discussion on p. 10); if J^{ℓ} is not less than δ, the sequence is unaltered. Then after setting σ to the starting value σ_1 the program execution returns to that part which is common to all loops, the output portion. The program then starts the next iteration exactly as it does in the reset loop.

If the test of the metric had resulted in a failure, then the next step would be to determine if the initial search strategy or the normal search strategy should be selected. (See discussion on initial search strategy.) If a "first success" has not occurred $p(0)$ will be selected, as shown on the flowgraph, from a uniform distribution. On the other hand, if a "first success" has occurred the normal strategy is selected. In this strategy the primary goal is to generate a new set of adjoint initial conditions using a gaussian noise source. However, before the actual generation of $p^{k+1}(0)$, certain of the algorithm parameters must be examined. These parameters control the maximum number of iterations at a given σ level, the changing of σ levels, and whether or not to start the search anew. First, a test is performed to find out if q consecutive trials have resulted in failures. If not, σ remains fixed and the program executes the jump ahead shown; if true, σ is incremented to the next value in the sequence $(\sigma_1, \ldots, \sigma_\gamma)$. Next is a test to determine if $\sigma > \sigma_\gamma$. If not, a jump ahead is executed. If it is, then a test is made to determine if the sequence has been gone through C times. If so, the program will reinitialize itself by entering the reset loop for a new start. This does not put the hybrid system in reset state but only restarts the algorithm. If the program did not reinitialize, then σ is set to σ_1, to prepare it for another cycle through the sequence.

This brings the execution of the program to the generation of the noise vector ξ^{k+1} (where it would have been if there were less than q consecutive failures or if σ was not greater than σ_γ). The distribution of the noise generator is gaussian with zero mean (see eq. (6)). Finally, the new adjoint initial conditions $p^{k+1}(0)$ are obtained by adding ξ^{k+1} to the last set of adjoint initial conditions $p^{\ell}(0)$ that produced a success, and the program moves to the output section.

The search mode continues reducing the metric by the selection of $p(0)$ until a value is found which gives a solution to the simulated control problem. This solution is sensed at the point in the program where J^{ℓ} is compared with ε. If the test indicates that a solution has been found then a branch to the left is executed where the first operation is to set up the end-state loop condition. Next, the data representing a solution are logged on the printer and the new adjoint initial conditions set to the value that gave the solution. The program then goes to output with the system set to perform the end-state loop.

End-state loop.- The end-state loop is shown in figure 29 by the line weight emphasis. In end state the integrators still sequence through their

initial condition and operate modes just as in the search mode. However, p(0) remains fixed at that value which produced the solution. In this manner it is possible to observe the optimal solutions found by the search algorithm on the CRT display described earlier.

Analog Program, Fixed Time

The analog program for fixed-time problems consists of the implementation of the state, adjoint and control equations. The text contains specific examples of nonlinear high-order differential equation sets encountered in a simulation of this type. As programming such equation sets on the analog computer is quite conventional, program diagrams were not considered essential to this report. However, one uncommon feature in programming these equations should be pointed out. Ordinarily, one can make a reasonable estimate of the range of state variables, and so forth, such that good scaling can be chosen. For the random-search method, however, the scaling is chosen to allow some variation from the expected solution. Nevertheless, on a moderate number of iterations the time histories of the state and adjoint variables will exceed the voltage range of the analog computer. Hence precaution must be taken for any saturation that may occur.

Analog Program, Free Time

The analog program of a free-time problem is essentially the same as that used for a fixed-time problem. There are two exceptions as noted earlier. First, the boundary value metric $J(t)$ must be computed continuously (as a function of the state variables) throughout the specified time interval $[0,T]$. For practical reasons the computation should be done on the analog machine. In contrast, for the fixed-time problem $J(t)$ is computed (as a function of the state variables) at a fixed point in time, that is, at $t = T$, the end time. This latter computation could and was done efficiently on the digital computer. Second, as was discussed in a previous part of this report it is necessary to determine the minimum value of the vector metric $J(t)$ over the interval $[0,T]$. The method of implementing this is shown in block diagram form in figure 30. This circuit computes the variable $J(\tau_m)$ defined as follows:

$$J(\tau_m) \equiv \min_{\tau} J(\tau) \qquad 0 < \tau < t$$

where τ_m is the value of τ for which the minimum is attained. The operation of this circuit can be briefly described as follows: If $J(t)$ is decreasing and is less than $J(\tau_m)$ (i.e., each component is less), then $J(\tau_m)$ is tracking or following identically $J(t)$; otherwise, $J(\tau_m)$ remains constant. In other words, $J(\tau_m)$ remains constant at the least value of $J(t)$ to date in the continuous computation of $J(t)$. Finally when $t = T$ we can see that we will have our desired result stored in memory, namely, $J(t_m)$ where

$$J(t_m) \equiv \min_{t} J(t) \qquad 0 \leq t \leq T$$

and t_m is the value of t at which the minimum is attained. Note that we do not explicitly use the value of τ_m or t_m in the iterative scheme. However, when $J(t_m)$ has been successfully reduced to within the ε region it is an easy matter to read out the value of t_m (referred to as t_{ms}) which gave the solution.

Although it would seem that the analog equipment required to track the necessary signals at the high repetitive frequencies would be a critical limitation to this approach it was found to work quite well in the satellite problem discussed in the text.

Digital Flowgraph, Free Time

Figure 31 is a flowgraph for free-time problems and autonomous systems. It is the same as the flowgraph of figure 25 for the fixed-time problem except for the following two modifications indicated in figure 31 by shaded boxes. First, for reasons discussed previously, the computation of the metric $J(t)$ must be done continuously on the analog machine thereby making the computation on the digital superfluous. The digital machine simply needs $J(t_m)$ from the analog machine, whereas in the fixed-time case the states $x(T)$ were read and $J(T)$ computed. These modifications are shown in the shaded region of the upper portion of figure 31. Second, the shaded region in the lower portion of this same figure reflect the changes necessary to ensure that the Hamiltonian be zero. This means computationally that all of the components of the vector $p(0)$ except one (shown here as $p_1(0)$) are chosen randomly as before, but the remaining component is determined from an algebraic equation so that $H = 0$ as required by the theory. Implementation for free time in the program does not appreciably affect the time for one iteration.

REFERENCES

1. Knapp, Charles H.; and Frost, Paul A.: Determination of Optimal Control and Trajectories Using the Maximum Principle in Association With a Gradient Technique. Joint Automatic Control Conference, Stanford University, June 24-26, 1964, pp. 222-227.

2. Brooks, S. H.: A Discussion of Random Methods for Seeking Maxima. Operations Research, vol. 6, no. 2, March 1958.

3. Karnopp, D. C.: Random Search Techniques for Optimization Problems. Automatica, vol. 1, Pergamon Press, 1963, pp. 111-121.

4. Bocharov, I. N.; and Fel'dbaum, A. A.: An Automatic Optimizer for the Search for the Smallest of Several Minima. (A Global Optimizer) Automation and Remote Control (trans. of Avtomatika i Telemekhanika), vol. 23, no. 3, March 1962.

5. Bekey, G. A.: Parameter Optimization by Random Search Using Hybrid Computer Techniques. AFIPS Conference Proceedings, vol. 29, 1966.

6. Favreau, R. R.; and Franks, R. G.: Statistical Optimization. Proc. 2nd International Analogue Computation Meetings, 1958.

7. Maybach, R. L.: Solution of Optimal Control Problems on a High-Speed Hybrid Computer. Simulation, vol. 7, no. 5, Nov. 1966.

8. Stewart, Elwood C.; Kavanaugh, William P.; and Brocker, David H.: Study of a Global Search Algorithm for Optimal Control. Presented at the Fifth International Analogue Computation Meetings, Lausanne, Switzerland, Aug. 28-Sept. 2, 1967, Journees Internationales de Calcul Analogique, vol. 1.

9. Kavanaugh, William P.; Stewart, Elwood C.; and Brocker, David H.: Optimal Control of Satellite Attitude Acquisition by a Random Search Algorithm on a Hybrid Computer. Presented at the 1968 Spring Joint Computer Conference, Atlantic City, New Jersey, April 30-May 2, 1968, AFIPS Conference Proceedings, vol. 32, 1968, pp. 443-452.

10. Pontryagin, L. S.; Boltyanskii, V. G.; Gamkrelidze, R. B.; and Mischenko, E. F.: The Mathematical Theory of Optimal Processes. Interscience Pub., N. Y., 1962.

11. Athans, Michael; and Falb, Peter L.: Optimal Control. McGraw-Hill Book Co., 1966.

12. Rozonoer, L. I.: L. S. Pontryagin's Maximum Principle in the Theory of Optimal Systems I, II, III. Automation and Remote Control, vol. 20, Oct., Nov., Dec. 1959, pp. 1517-1532.

13. Zadeh, L. A.: Optimality and Non-Scalar-Valued Performance Criteria. IEEE Transactions on Automatic Control, vol. AC-8 no. 1, Jan. 1963, pp. 59-60.

14. Sabroff, A.; Farrenhopf, R.; Frew, A.; and Gran, M.: Investigation of the Acquisition Problem in Satellite Attitude Control. Tech. Rep. AFFDL-TR-65-115, Space Technology Labs, Wright-Patterson, Dec. 1965.

TABLE 1.- COMPARISON OF CONTROL SYSTEMS

Control system	Initial condition	Fuel
No control	(0,0,1,-10,10,10)	0
Optimal proportional control (no saturation)	(0,0,1,-10,10,10)	.65
Optimal impulse control	(0,0,1,-10,10,10)	.28
Optimal nonlinear control (with saturation)		
Initial alinement	(0,0,1,-10,10,10)	.31
Initial nonalinement	$(0,1/\sqrt{2},1/\sqrt{2},-10,10,10)$	.43

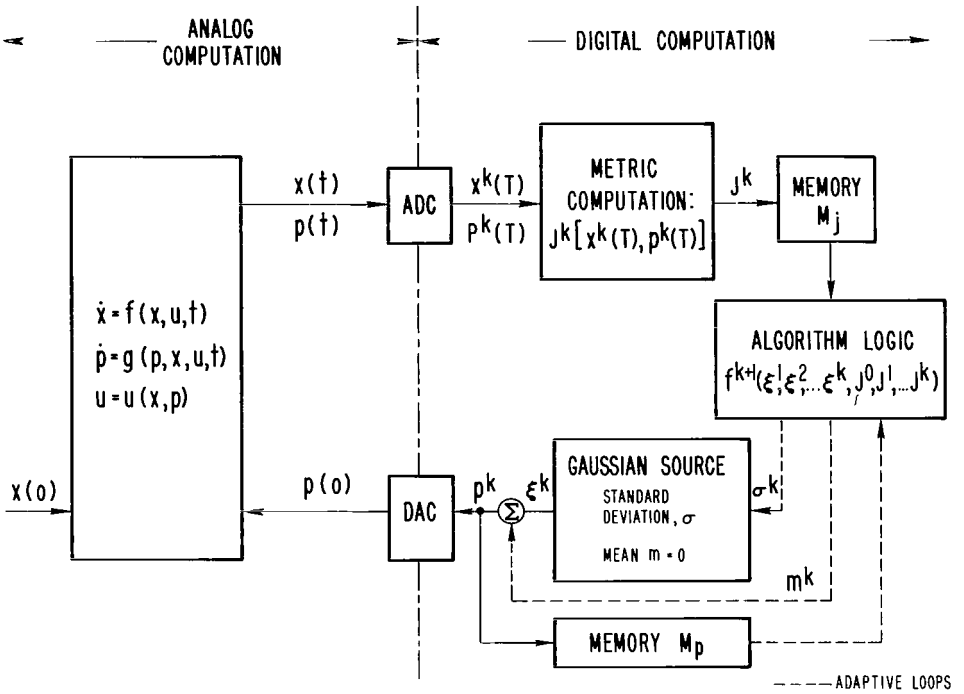


Figure 1.- Hybrid system block diagram of random search algorithm for fixed-time problems.

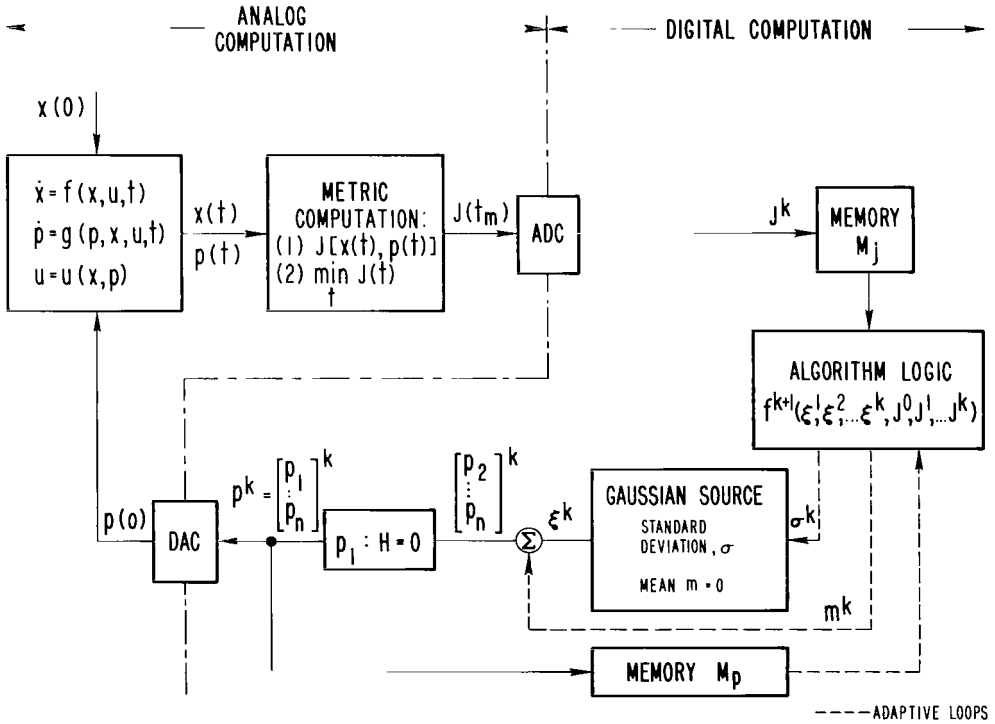


Figure 2.- Hybrid system block diagram of random search algorithm for free-time problems.

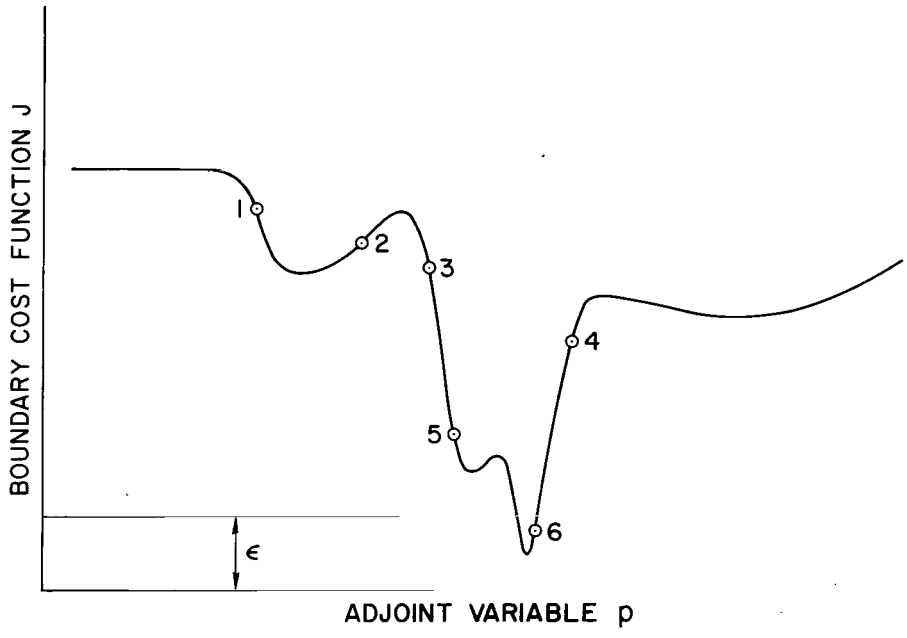


Figure 3.- Typical boundary cost function surface.

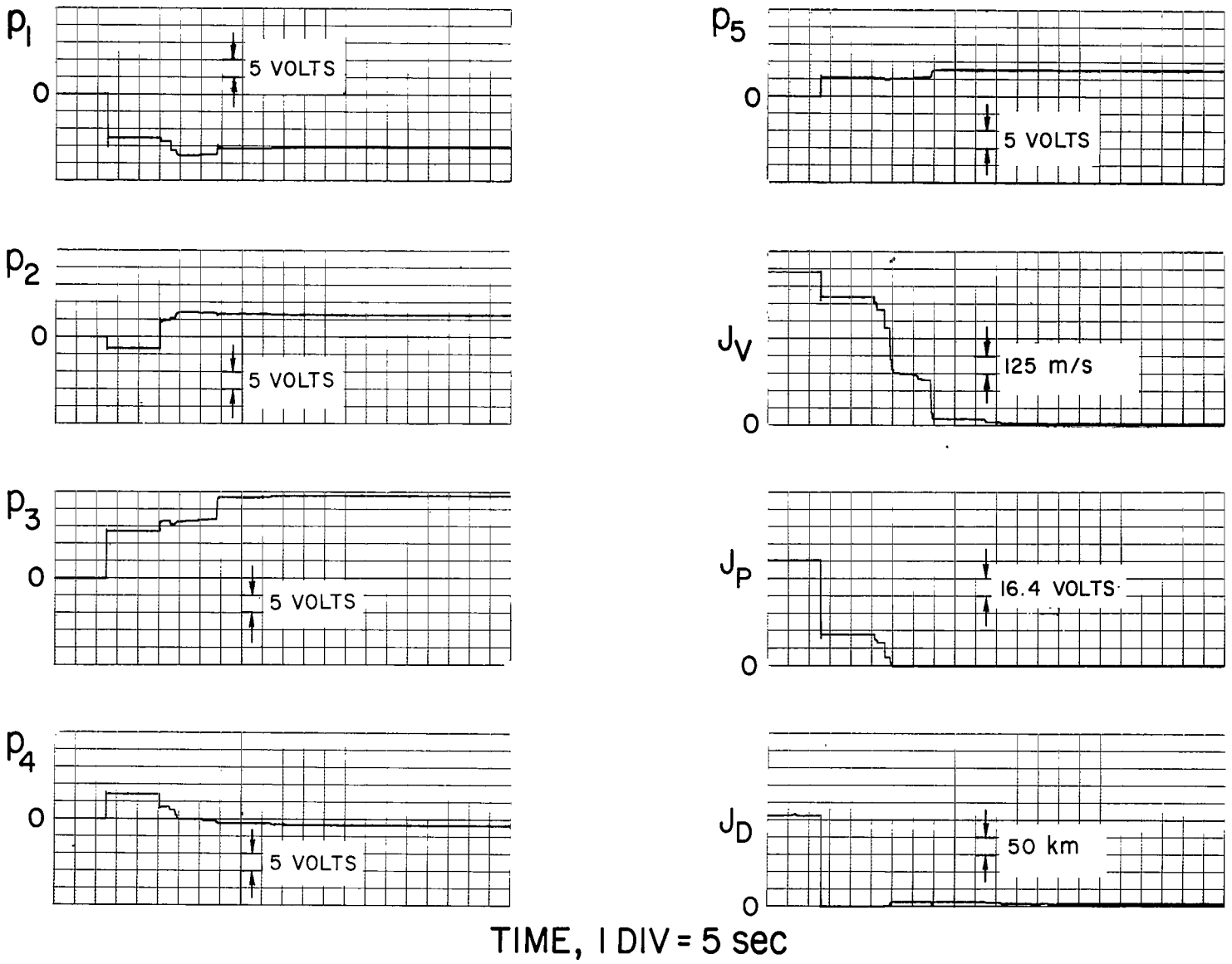


Figure 4.- Behavior of successful iterations during search.

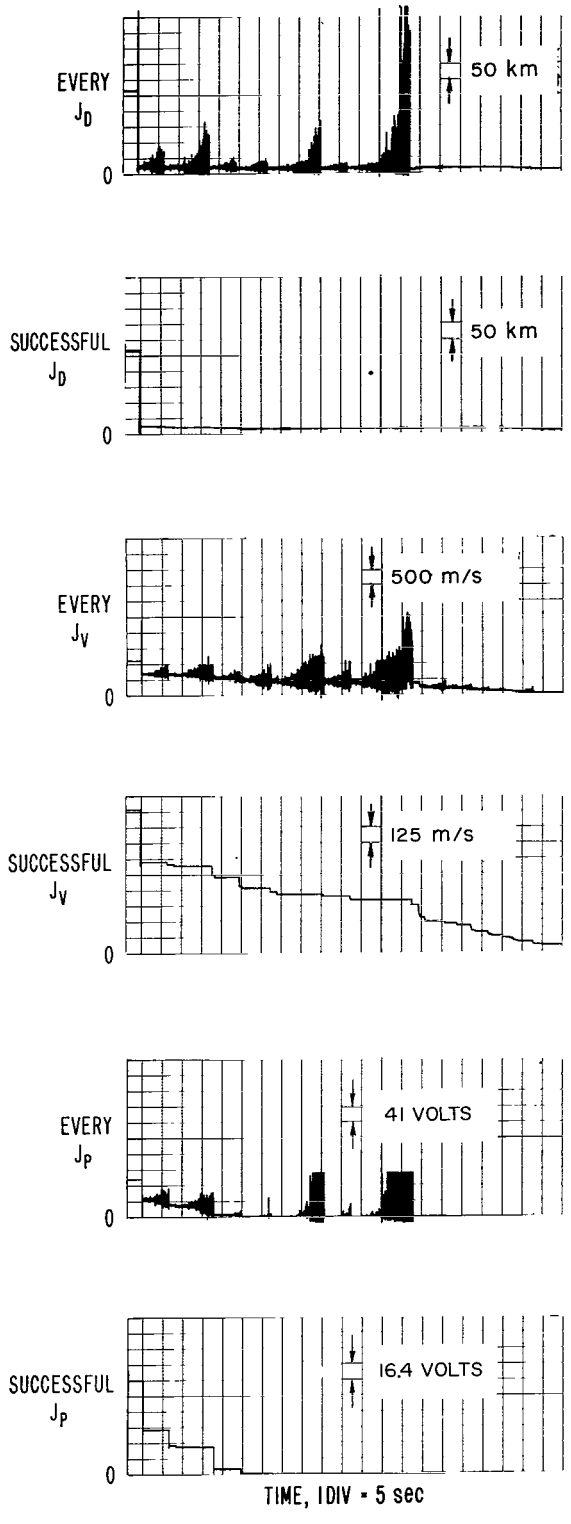


Figure 5.- Behavior of every iteration during search.

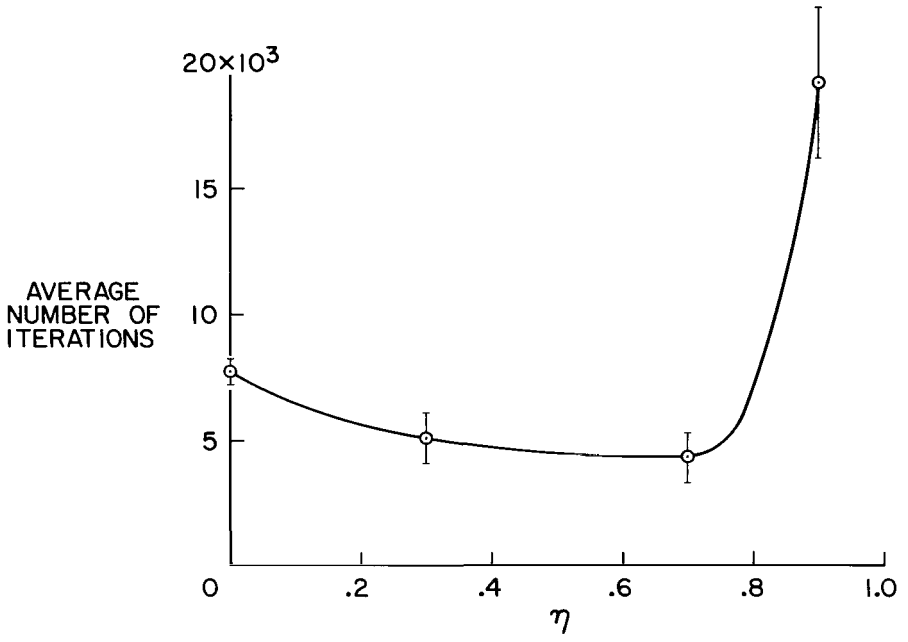


Figure 6.- Effect on convergence of threshold strategy.

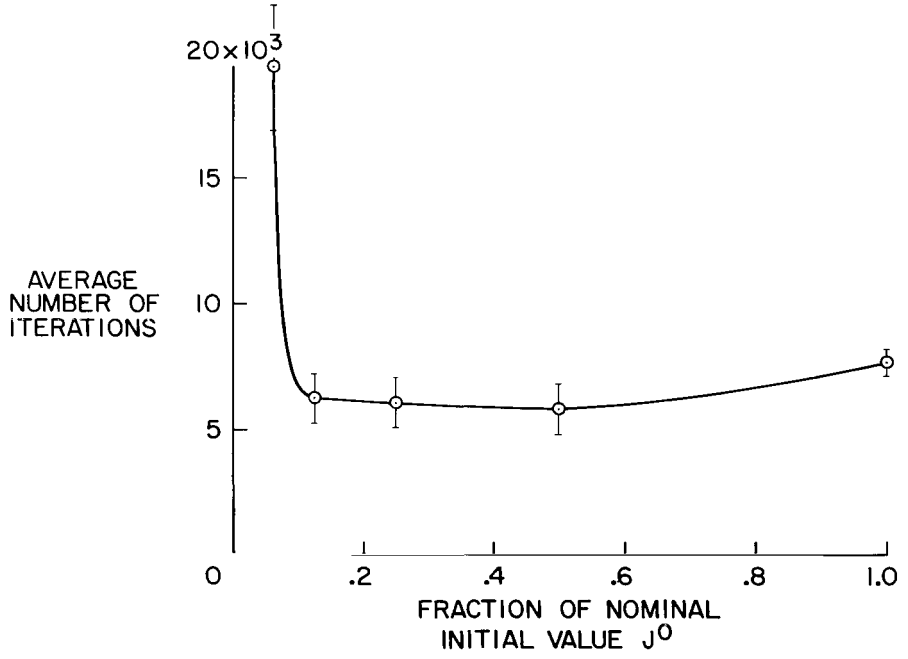


Figure 7.- Effect on convergence of varying initial value J°.

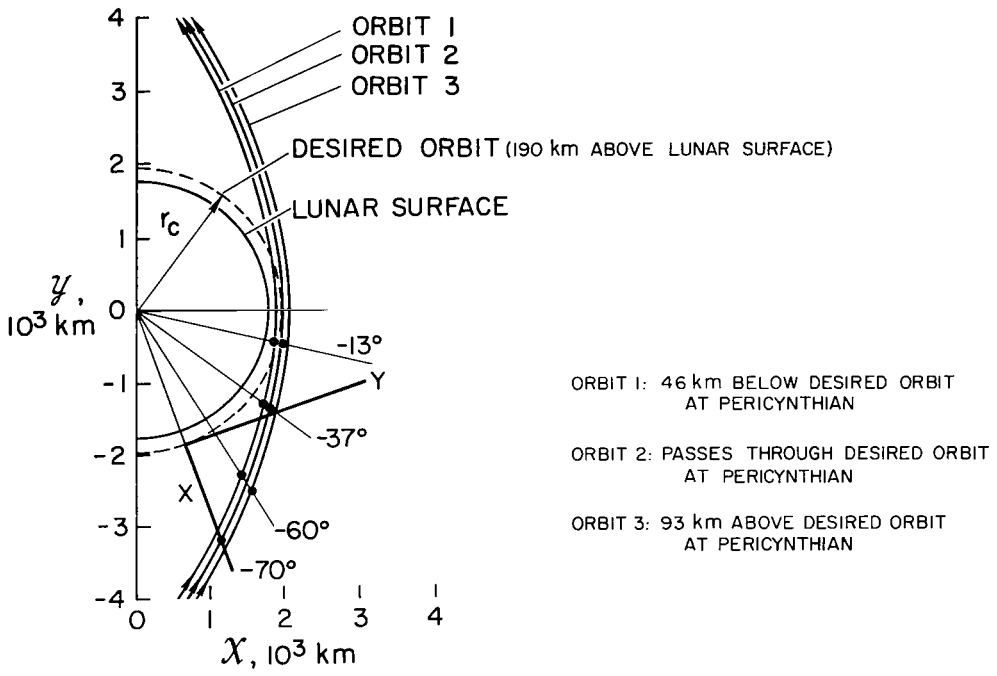
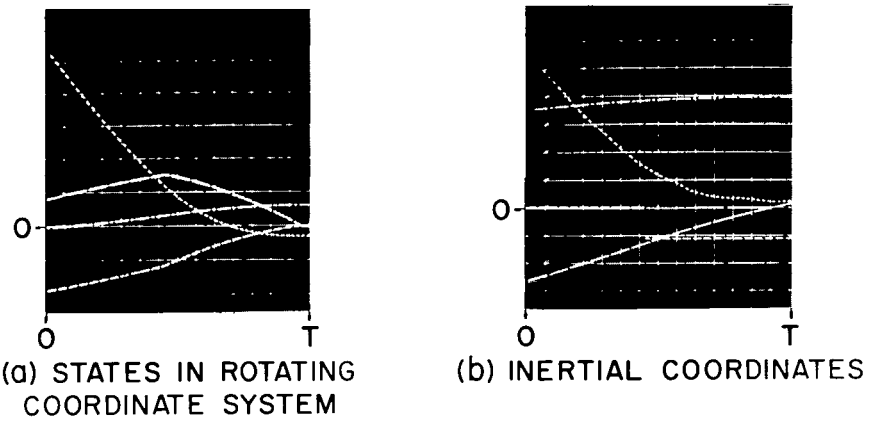


Figure 8.- Geometry of orbit-transfer problem.

----- x_1 I DIV = 50 km
--- x_2 I DIV = 500 m/s
—·— x_3 I DIV = 500 km
—— x_4 I DIV = 500 m/s

----- $r-r_c$ I DIV = 50 km
--- $-u_3$ I DIV = 25 kg/s
—·— x I DIV = 500 km
—— y I DIV = 500 km



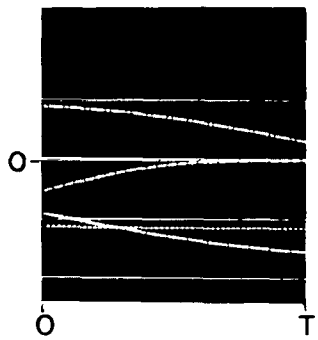
(a) STATES IN ROTATING COORDINATE SYSTEM

(b) INERTIAL COORDINATES

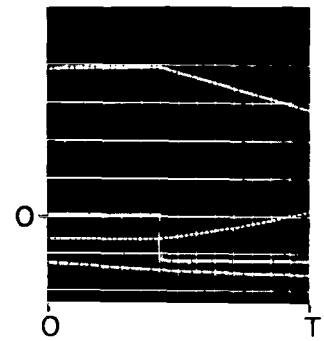
Figure 9.- Time history solutions; orbit 2, x(0) at -37°, T = 600 sec.

----- p_1 1 DIV = 10 volts
- - - p_2 1 DIV = 10 volts
— — — p_3 1 DIV = 1 volt
—— p_4 1 DIV = 10 volts

----- p_5 1 DIV = 10 volts
—— $-\|P\|$ 1 DIV = 10 volts
—·— x_5 1 DIV = 10,000 kg
—— $-u_3$ 1 DIV = 25 kg/s

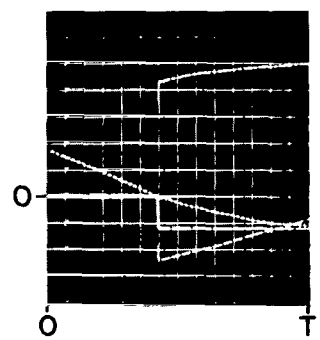


(c) ADJOINT STATES



(d) MISCELLANEOUS STATES

----- SWITCHING FUNCTION, v
— — — u_1 1 DIV = .2
—·— u_2 1 DIV = .2
—— $-u_3$ 1 DIV = 25 kg/s

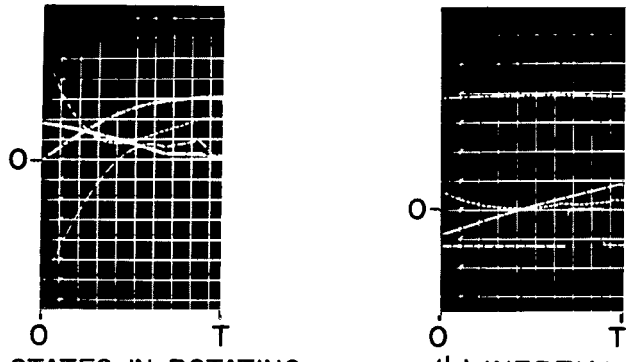


(e) CONTROL VECTOR

Figure 9.- Concluded.

----- x_1 I DIV = 5 km
--- x_2 I DIV = 50 m/s
—·— x_3 I DIV = 50 km
—— x_4 I DIV = 500 m/s

----- $r-r_c$ I DIV = 50 km
---- $-u_3$ I DIV = 25 kg/s
—·— x I DIV = 500 km
—— y I DIV = 500 km



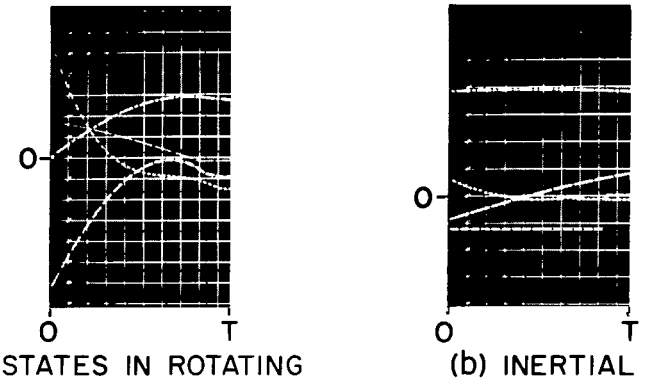
(a) STATES IN ROTATING COORDINATE SYSTEM

(b) INERTIAL COORDINATES

Figure 10.- Time history solutions 1; orbit 2, x(0) at -13°, T = 400 sec.

----- x_1 I DIV = 5 km
--- x_2 I DIV = 50 m/s
—·— x_3 I DIV = 50 km
—— x_4 I DIV = 500 m/s

----- $r-r_c$ I DIV = 50 km
---- $-u_3$ I DIV = 25 kg/s
—·— x I DIV = 500 km
—— y I DIV = 500 km



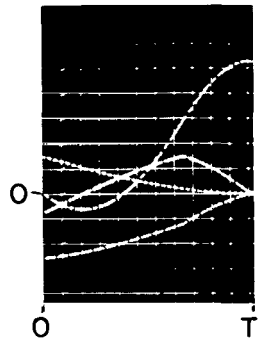
(a) STATES IN ROTATING COORDINATE SYSTEM

(b) INERTIAL COORDINATES

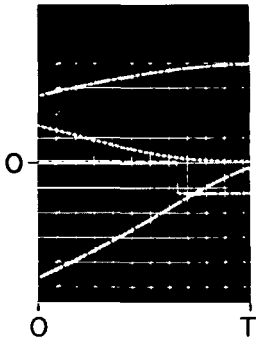
Figure 11.- Time history solutions 2; orbit 2, x(0) at -13°, T = 400 sec.

----- x_1 I DIV = 500 km
--- x_2 I DIV = 500 m/s
—·— x_3 I DIV = 50 km
—— x_4 I DIV = 500 m/s

----- $r-r_c$ I DIV = 500 km
--- $-u_3$ I DIV = 25 kg/s
—·— x I DIV = 500 km
—— y I DIV = 500 km



(a) STATES IN ROTATING COORDINATE SYSTEM

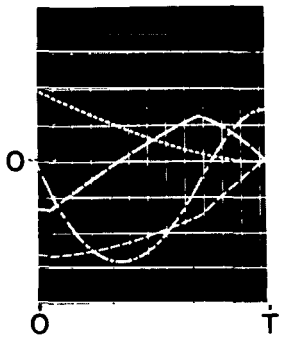


(b) INERTIAL COORDINATES

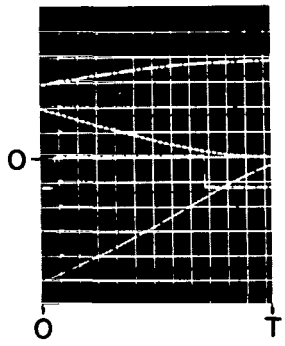
Figure 12.- Time history solutions; orbit 1, x(0) at -60°, T = 971 sec.

----- x_1 I DIV = 500 km
--- x_2 I DIV = 500 m/s
—·— x_3 I DIV = 50 km
—— x_4 I DIV = 500 m/s

----- $r-r_c$ I DIV = 500 km
--- $-u_3$ I DIV = 25 kg/s
—·— x I DIV = 500 km
—— y I DIV = 500 km



(a) STATES IN ROTATING COORDINATE SYSTEM



(b) INERTIAL COORDINATES

Figure 13.- Time history solutions; orbit 3, x(0) at -60°, T = 1075 sec.

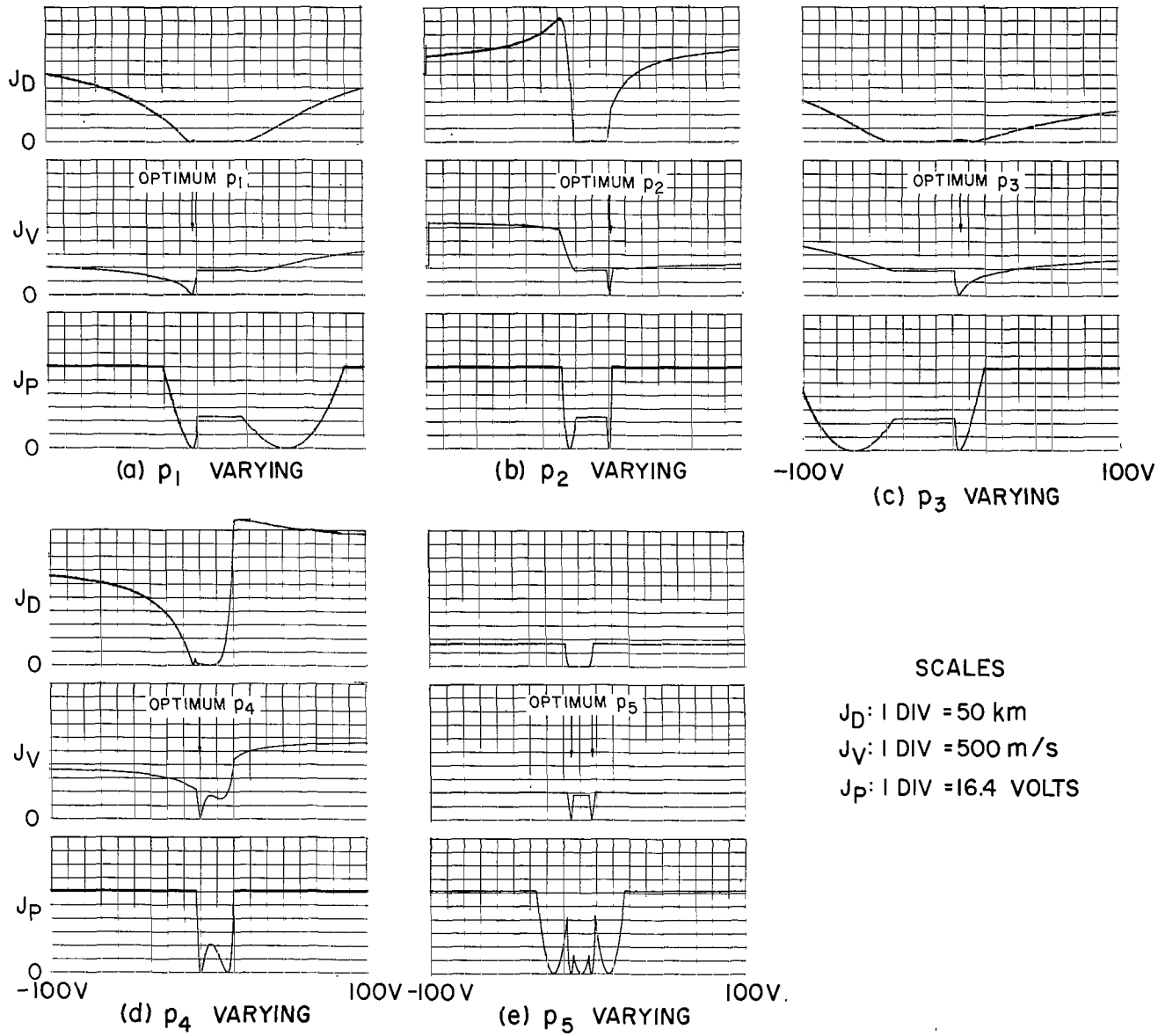


Figure 14.- Two-dimensional cross sections of boundary hypersurface; orbit 2, x(0) at -37°, T = 600 sec.

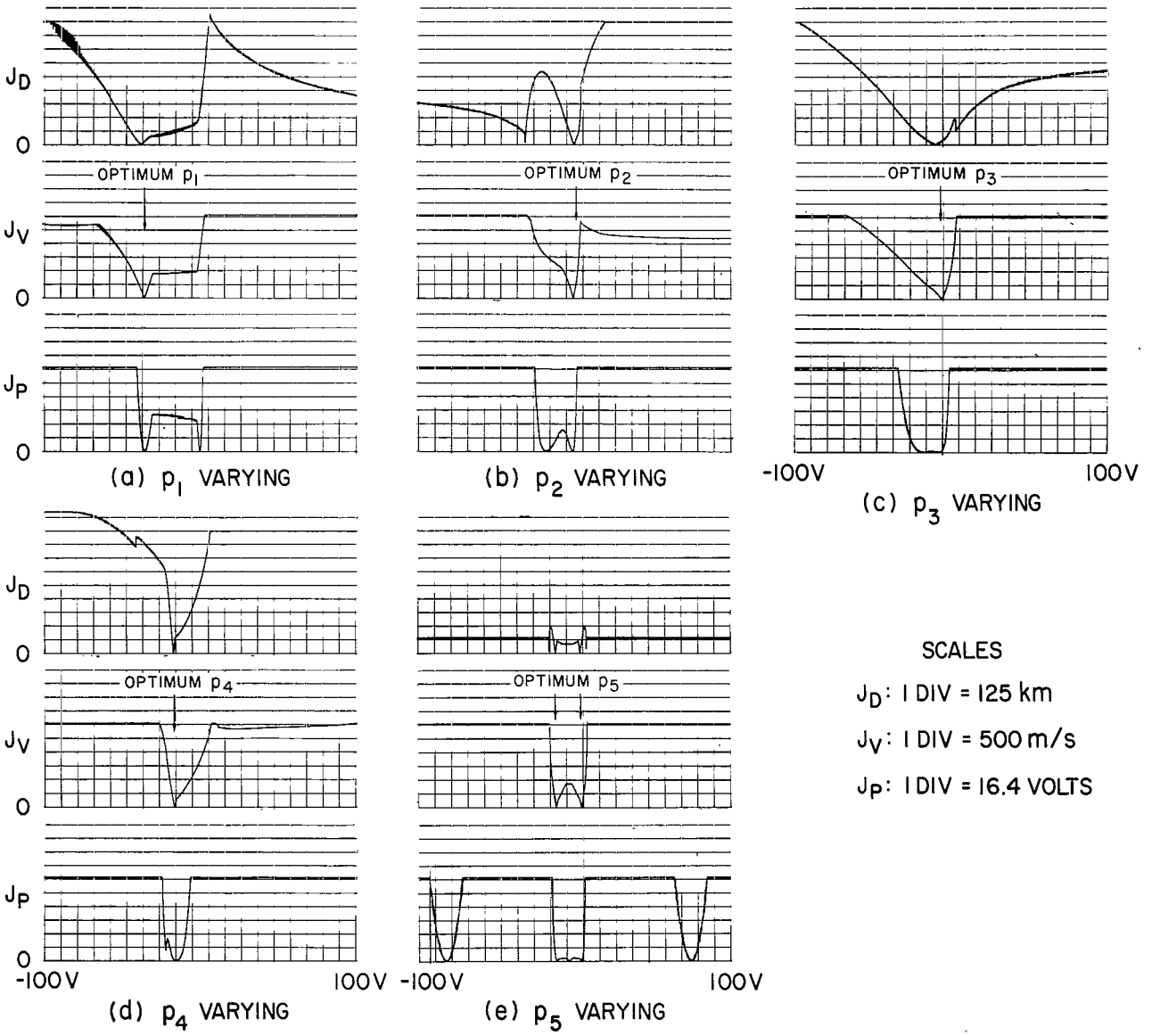
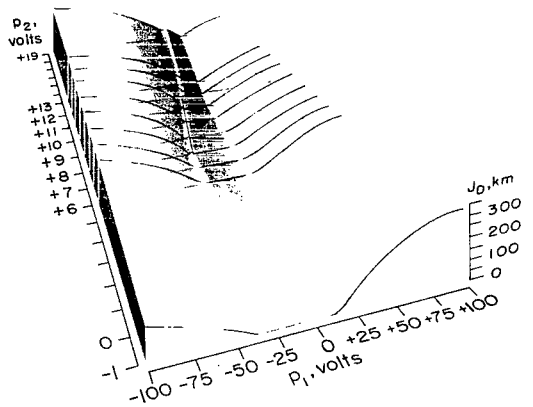
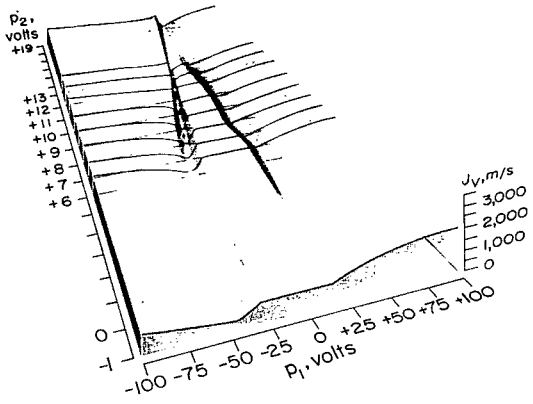
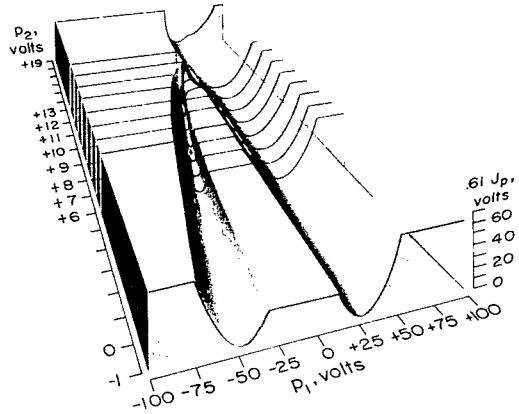


Figure 15.- Two-dimensional cross sections of boundary hypersurface; orbit 3, x(0) at -60°, T = 1075 sec.



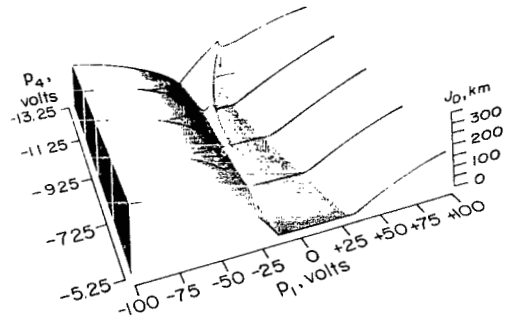


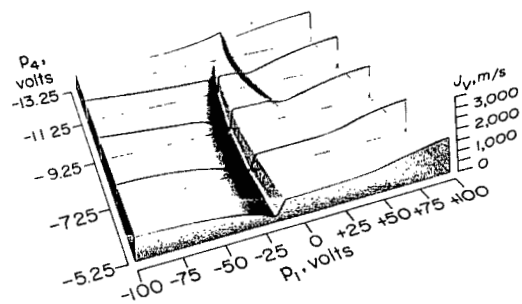
➡ INDICATES SOLUTION POINT



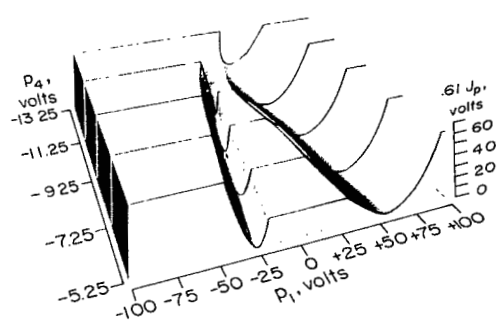
(a) p_1, p_2 plane.

Figure 16.- Three-dimensional cross sections of boundary hypersurface; orbit 2, x(0) at -37°, T = 600 sec.





► INDICATES SOLUTION POINT



(b) p_1, p_4 plane.

Figure 16.- Concluded.

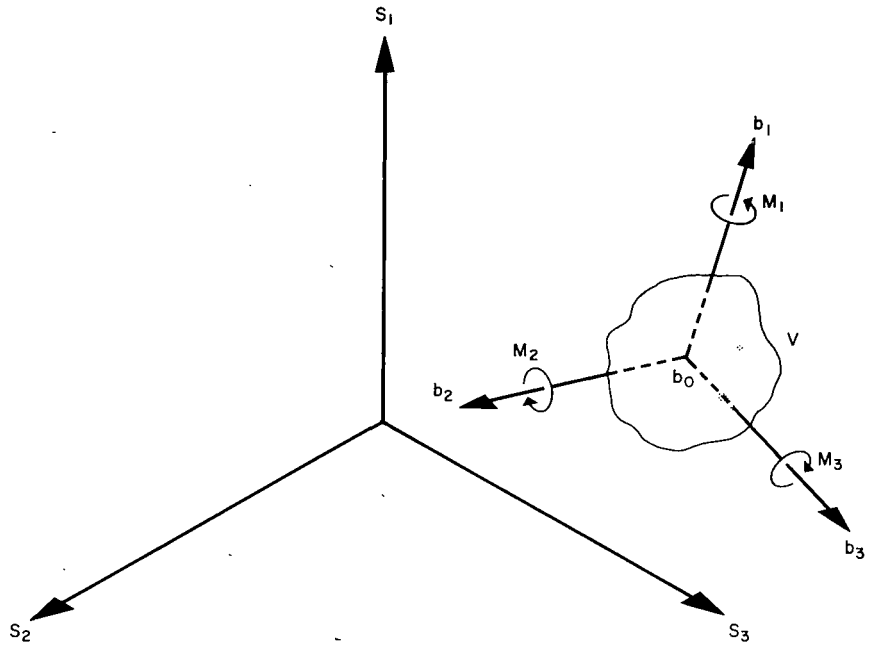


Figure 17.- Geometry of satellite attitude acquisition.

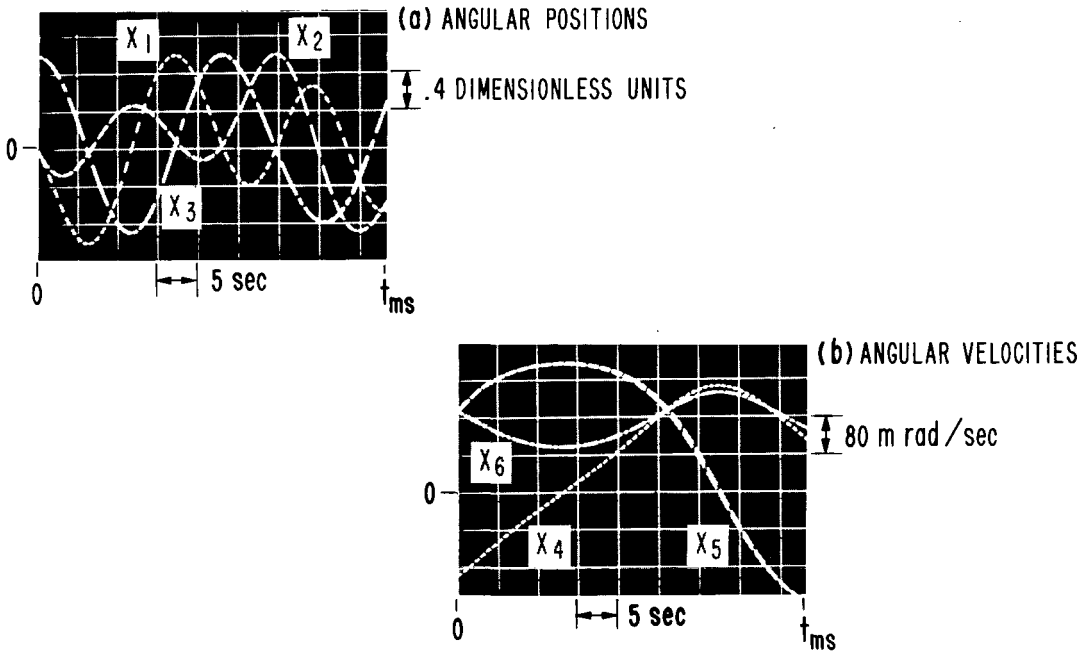


Figure 18.- Time history of state vector; no control; initial alinement of axes.

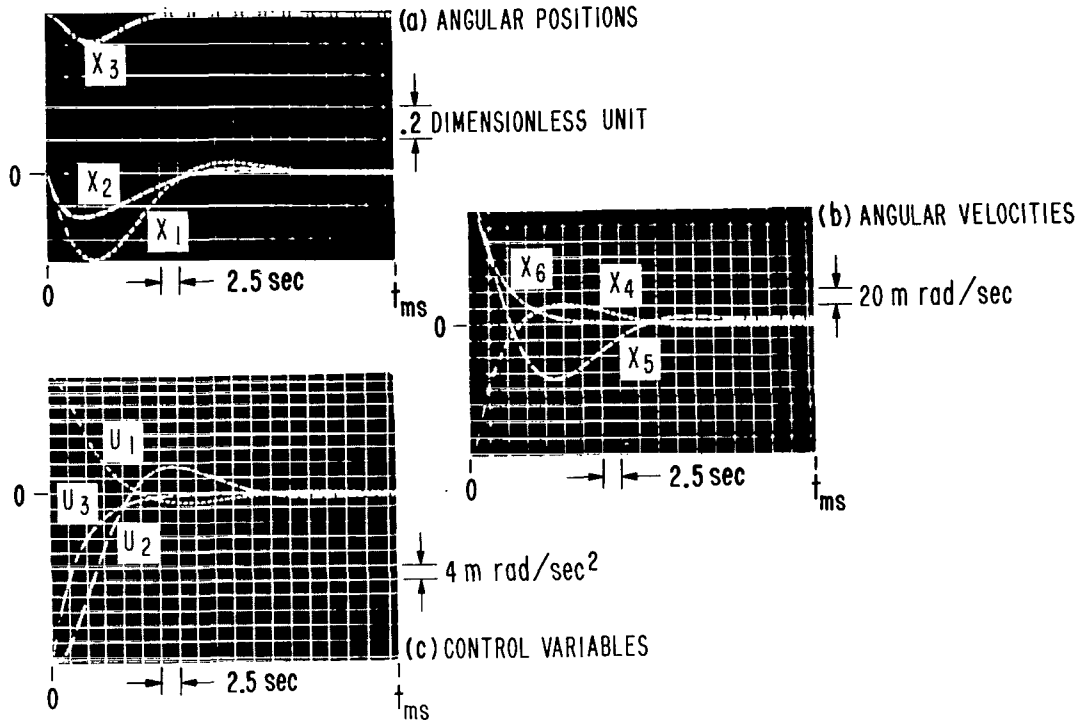


Figure 19.- Time histories of state and control vectors; optimal proportional control; initial alinement of axes.

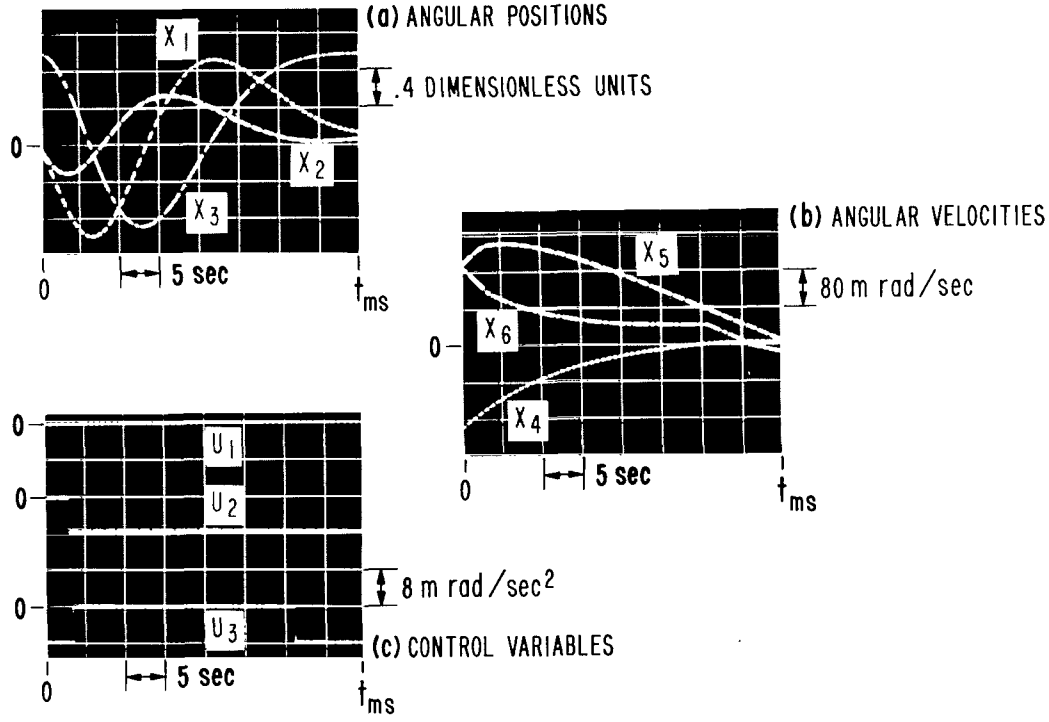


Figure 20.- Time histories of state and control vectors; optimal nonlinear control; initial alinement of axes.

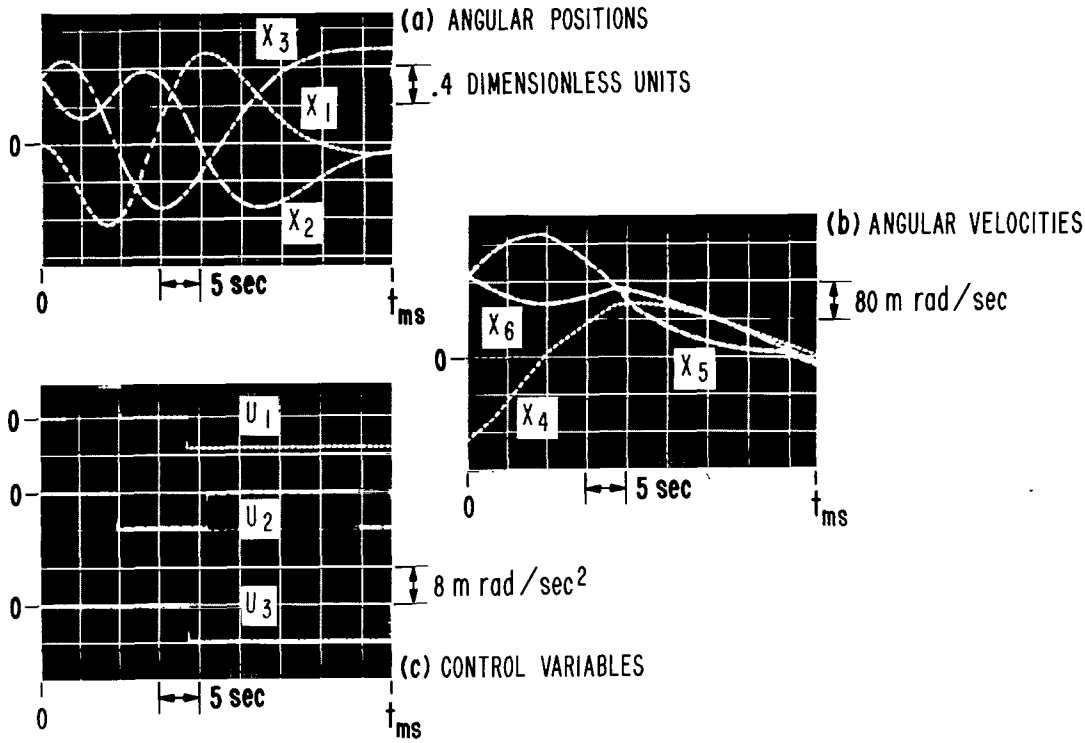


Figure 21.- Time histories of state and control vectors; optimal nonlinear control; initial nonalinement of axes.

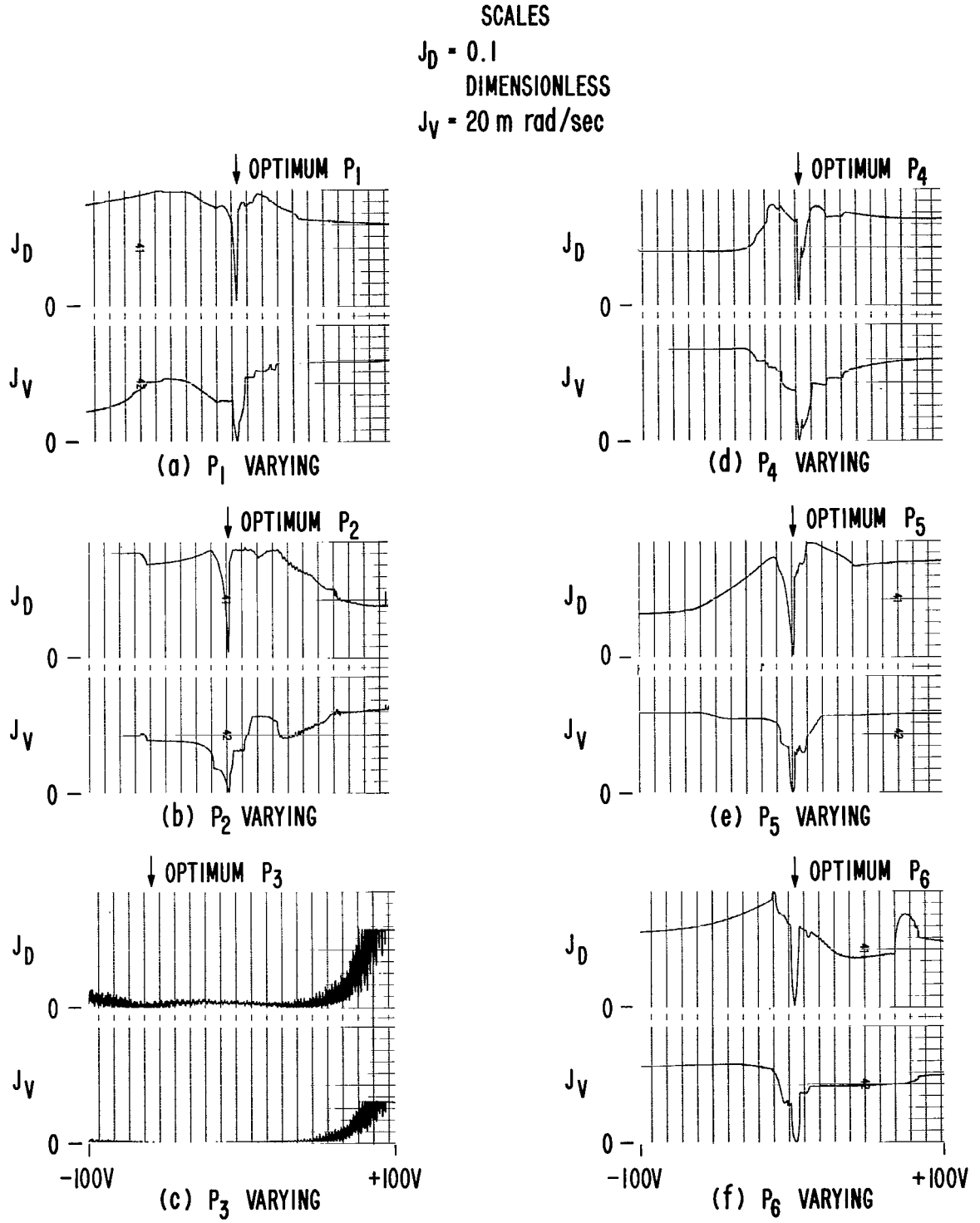


Figure 22.- Example two-dimensional cross sections of boundary hypersurfaces.

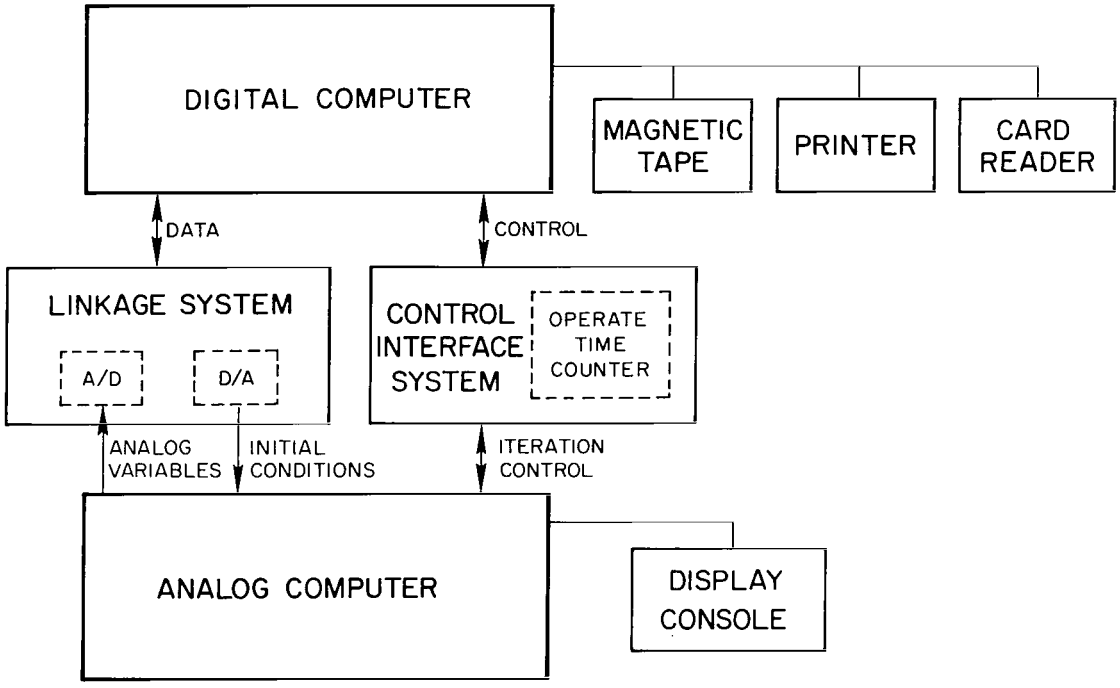


Figure 23.- Hybrid system hardware.

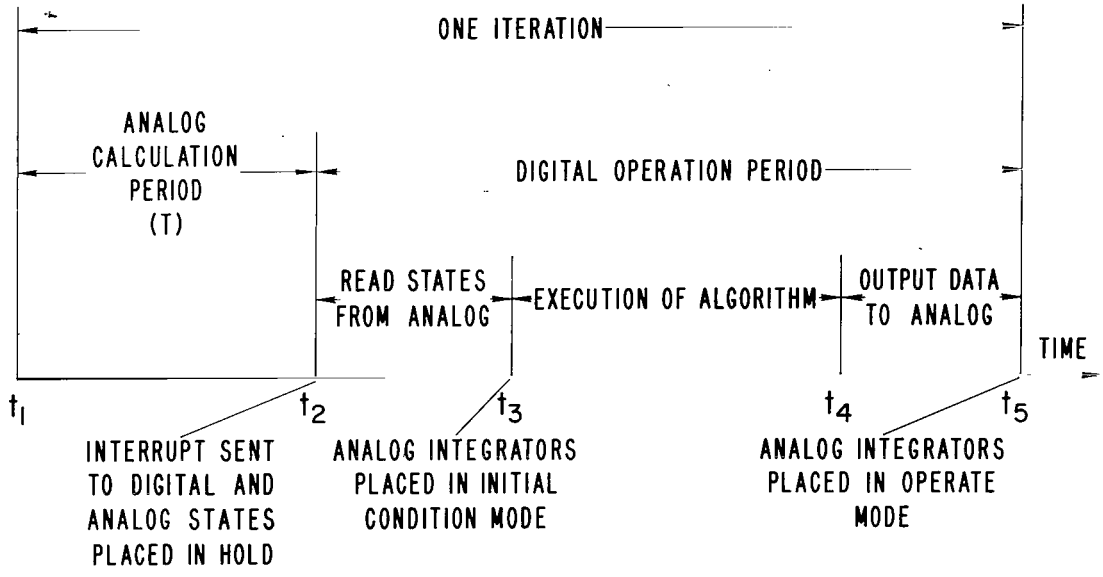


Figure 24.- Sequencing of events during one iteration.

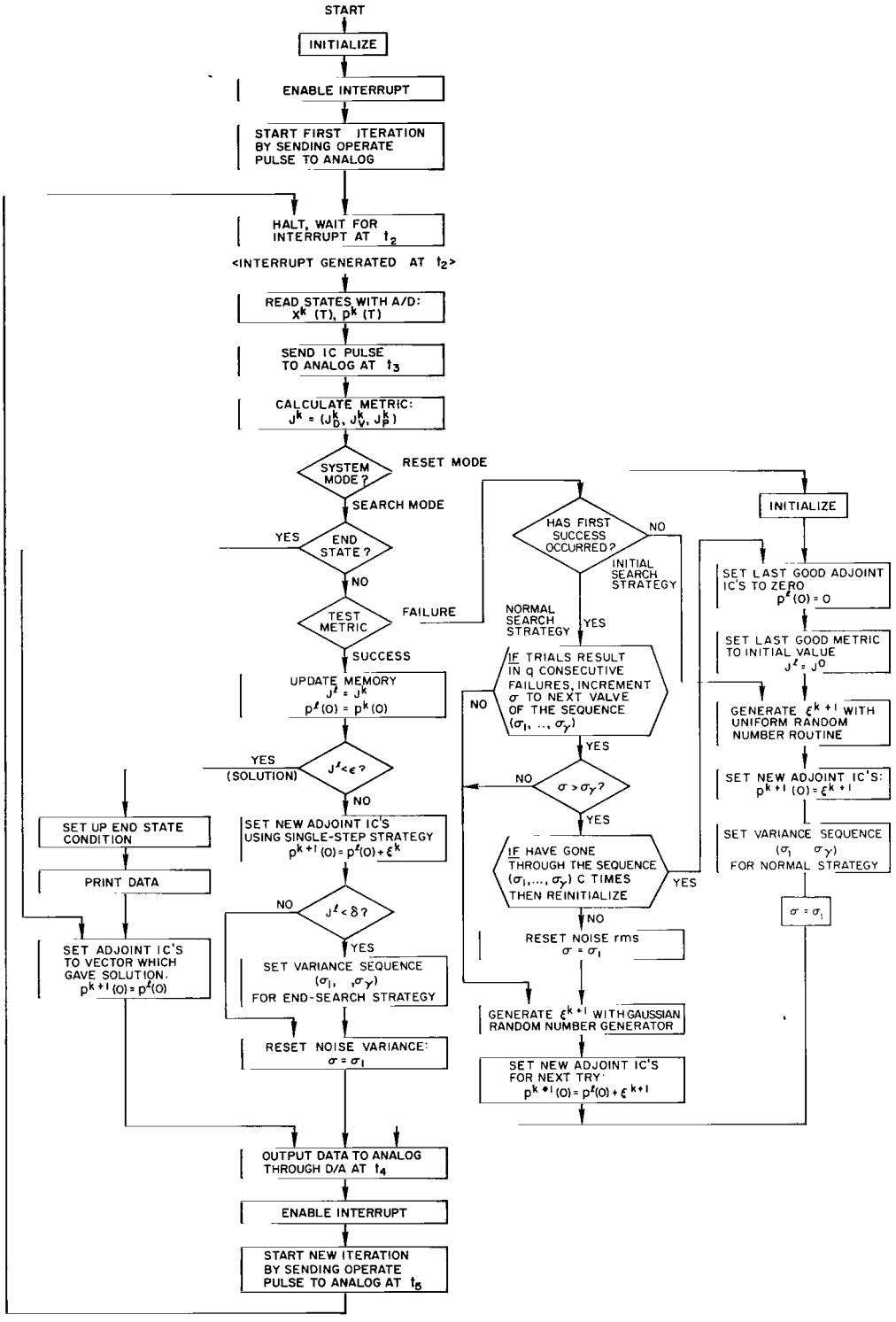


Figure 25.- Algorithm flow graph (simplified) for fixed-time problems.

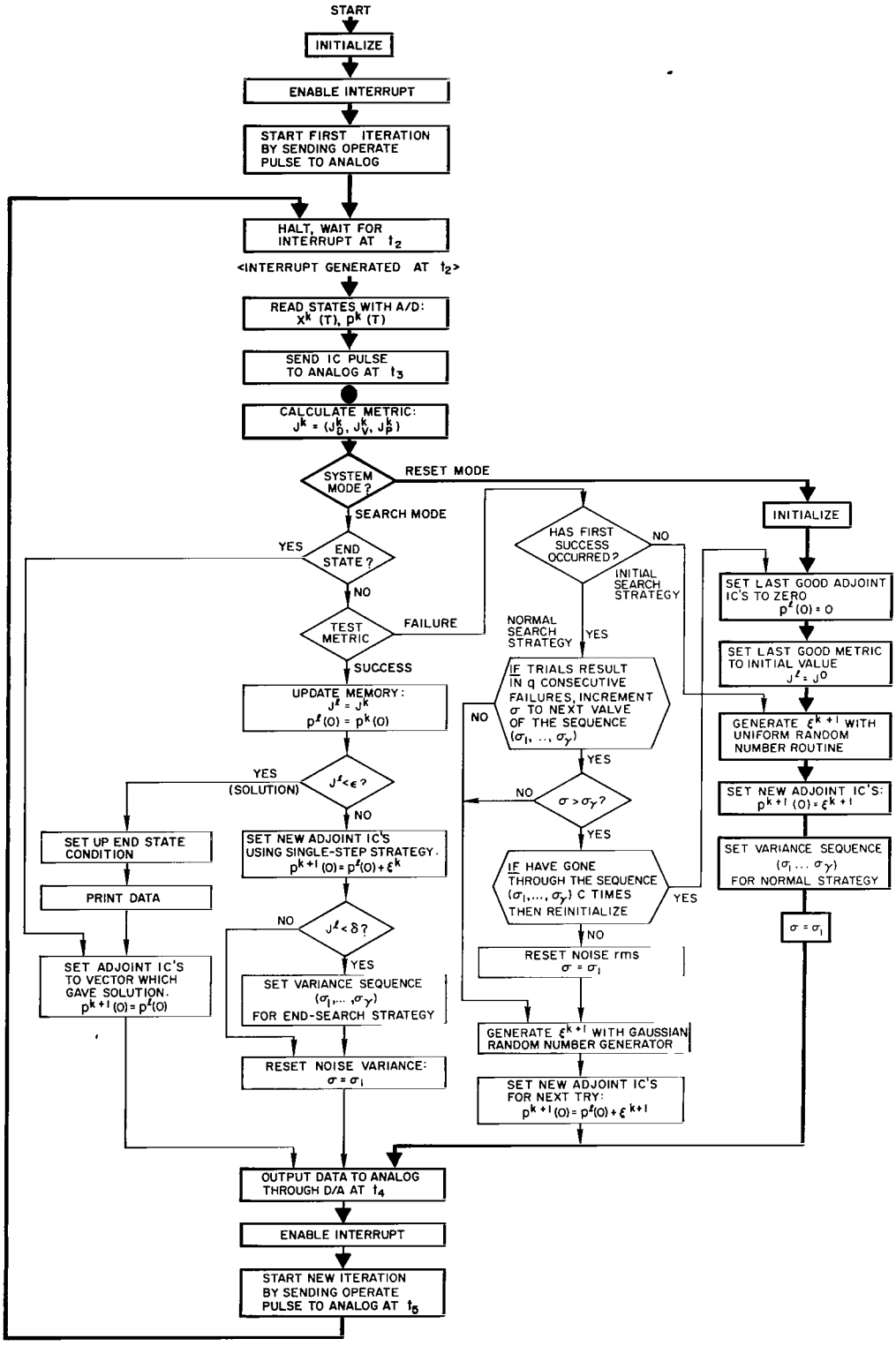


Figure 26.- Reset loop (simplified) for fixed-time problems.

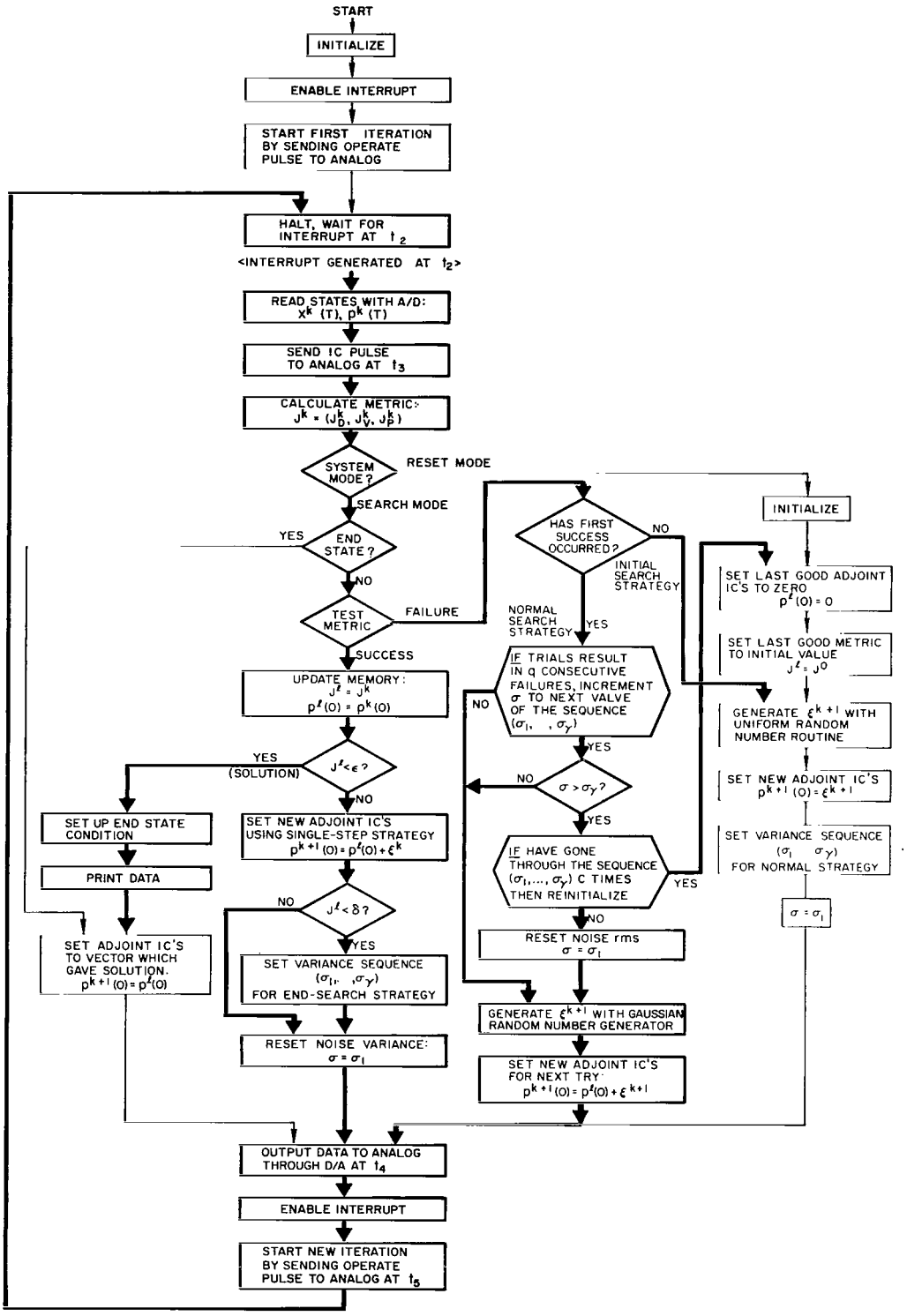


Figure 27.- Search loop (simplified) for fixed-time problems.

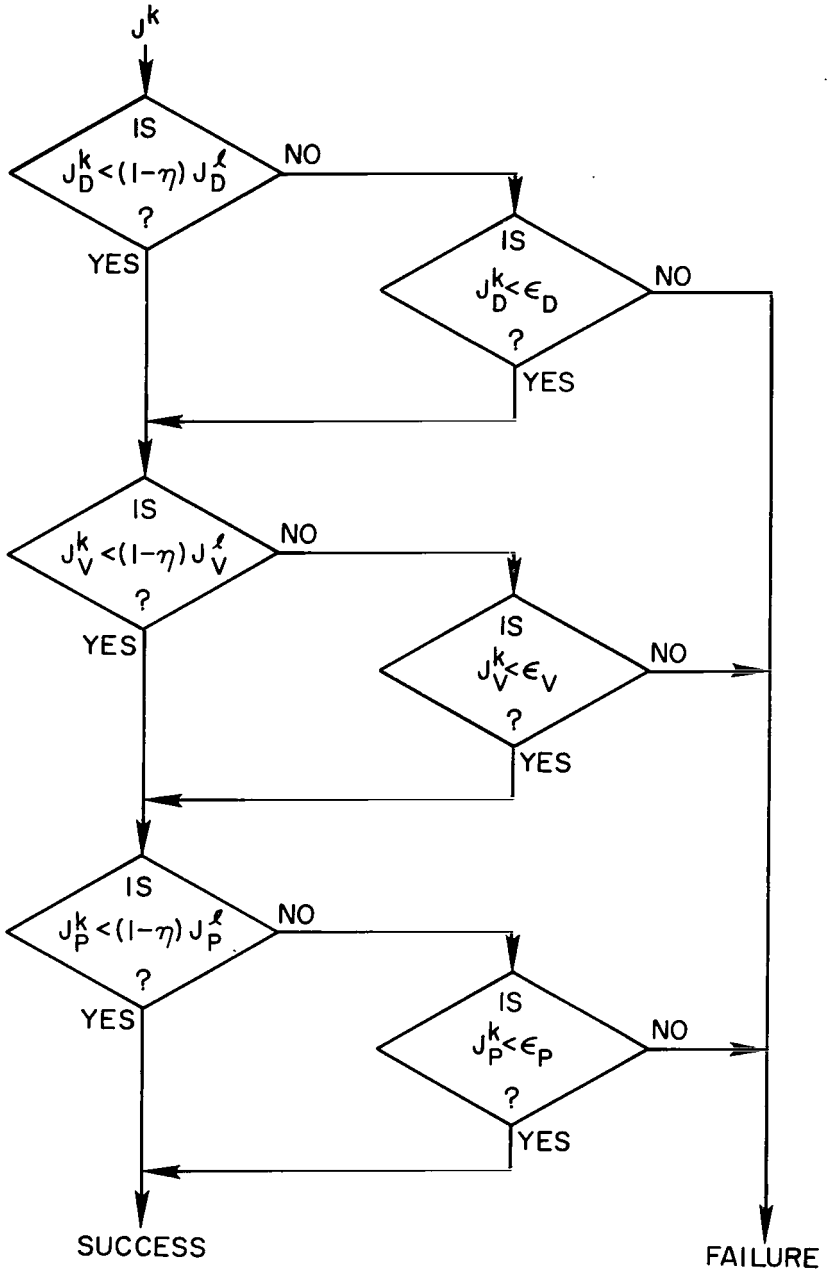


Figure 28.- Test-metric flow graph.

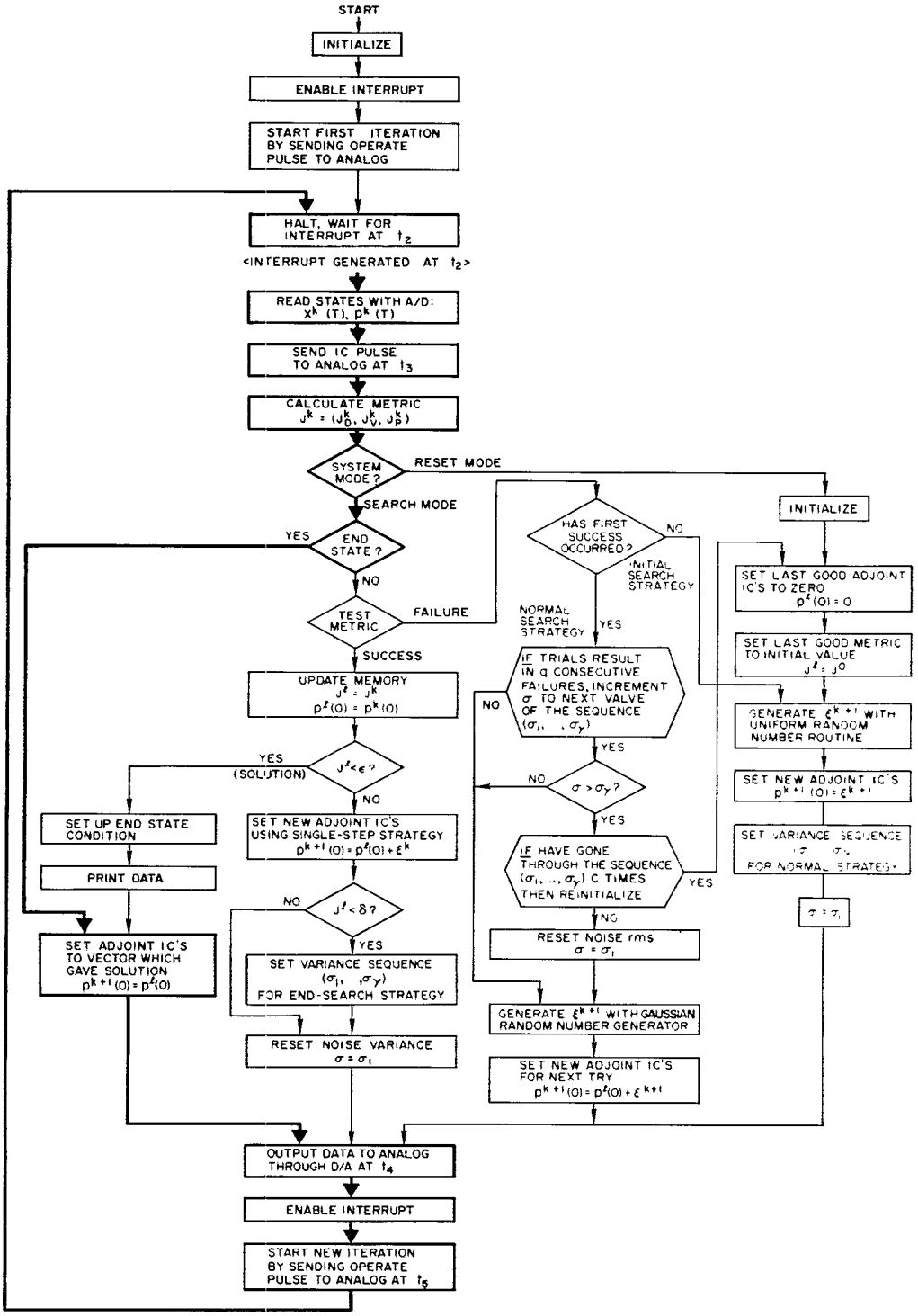


Figure 29.- End-state loop (simplified) for fixed-time problems.

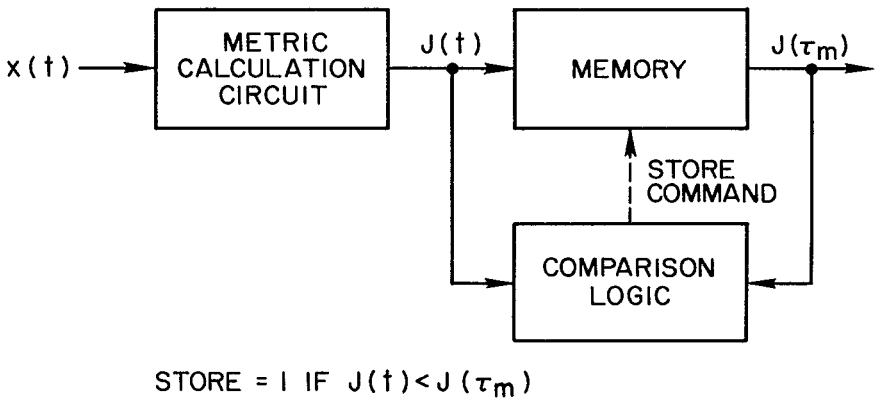


Figure 30.- Analog metric circuit for free-time problems.

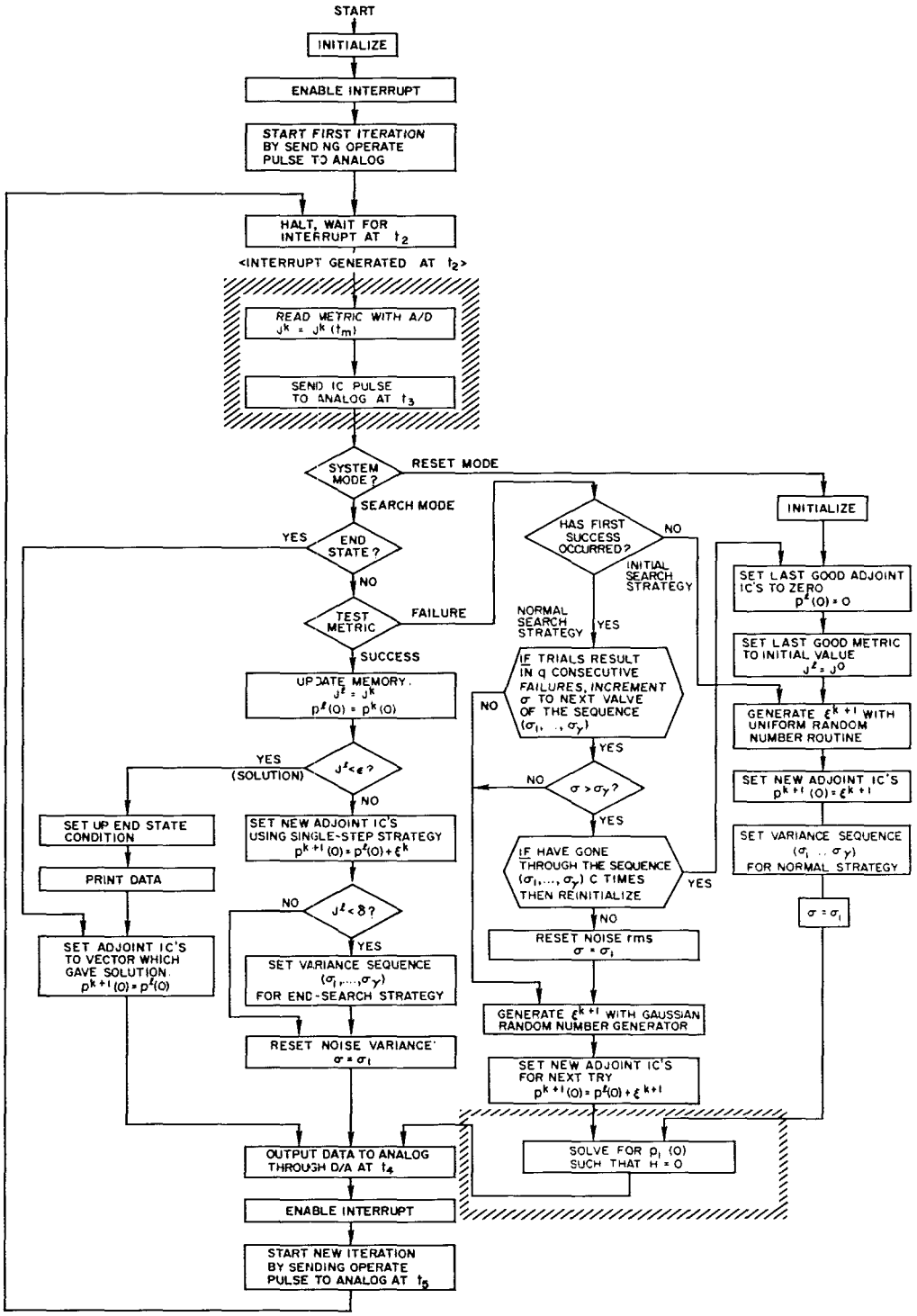


Figure 31.- Algorithm flow graph (simplified) for free-time problems and autonomous systems.

"The aeronautical and space activities of the United States shall be conducted so as to contribute . . . to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

—NATIONAL AERONAUTICS AND SPACE ACT OF 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

TECHNICAL REPORTS: Scientific and technical information considered important, complete, and a lasting contribution to existing knowledge.

TECHNICAL NOTES: Information less broad in scope but nevertheless of importance as a contribution to existing knowledge.

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons.

CONTRACTOR REPORTS: Scientific and technical information generated under a NASA contract or grant and considered an important contribution to existing knowledge.

TECHNICAL TRANSLATIONS: Information published in a foreign language considered to merit NASA distribution in English.

SPECIAL PUBLICATIONS: Information derived from or of value to NASA activities. Publications include conference proceedings, monographs, data compilations, handbooks, sourcebooks, and special bibliographies.

TECHNOLOGY UTILIZATION PUBLICATIONS: Information on technology used by NASA that may be of particular interest in commercial and other non-aerospace applications. Publications include Tech Briefs, Technology Utilization Reports and Notes, and Technology Surveys.

Details on the availability of these publications may be obtained from:

SCIENTIFIC AND TECHNICAL INFORMATION DIVISION

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D.C. 20546